

# Lesson 1: Play and Pause Commands

**Note:** **Code Jumper** is the physical, touchable kit. You have probably already used Play Pods and the Hub with your teacher! There is a block-coding part of the application which you interacted with when connecting the hub and the command pods and turning the dials and knobs to change sounds! While CJ Threads is the Python coding section, changes made in CJ Threads will not be shown in Code Jumper (the physical block-coding component). This is because CJ Threads lets you do so many more awesome things than the original Code Jumper. For example, you are no longer limited to the 8 Play pods. With CJ Threads, you can add a near limitless array of Play commands in Python!

**In this lesson:** When being asked to type in text or expressions, do not include the quotes that surround the text or expression. The text or expression is formatted in bold and surrounded by quotes.

**In This Lesson:** You’ll be creating your first Python program using the**Play** and **Pause** commands in CJ Threads.

## Materials Needed

* Windows 10 or 11 device with Internet connection
* These instructions
* Audio capability (speakers/headphones)

## Concepts

* Sequencing
  + Play sounds in desired sequence
  + Add **Pause** commands in sequence
  + Fix sequencing issues
* Sound Sequences
* Editing values
  + Adding numeric values for **Play** and **Pause** commands
* Adding **Play** and **Pause** commands
* Introduction to indentation in Python
  + Indentation inside functions
* Debugging sequences

### Examples

Think back to [Lesson 2—Sequences and Algorithms](https://codejumper.com/downloads/htmls/lessons/en/Lesson02/Lesson_2.html) from the original Code Jumper lessons. Remember when you used “Twinkle, Twinkle” to learn about **sequencing**? Here is an example in Python that will accomplish the same thing.

This is a reviewing activity if you want to try creating “Twinkle, Twinkle” before moving on to our first new sound set, “Hello World!”

### Indentation

In Python, indentation is used to group a block of code under a function, loop, and if statements. To create a block of code that is part of a function, it should have a common level of indentation. While Python allows for different types of indentation, CJ Threads uses four spaces to specify 1 indentation level. Notice below that every line after “def Thread1():” is indented four spaces, which specifies that those commands will be executed as part of the Thread1 function. Using CJ Threads, the tab key will increase indentation and shift tab will decrease indentation.

**Twinkle, Twinkle Little Star (Sequencing)**

def Thread1():

global x;

sounds = SG.GetSoundGroupWithName("Twinkle, Twinkle")

player.Play(sounds, 1, 2)

player.Play(sounds, 2, 2)

player.Play(sounds, 3, 2)

player.Play(sounds, 4, 2)

### Activity 1: Hello World

**Learning Objectives**

* Add a **Play** command and enter both values (**Sound** and **Pitch**)
* Add a **Comment** command and enter comment (**Text**)
* Formatting
* Create correct sequence of sounds with **Comment** commands in appropriate places
* Use **Edit** to select **Sound Set**
* View **Sequence of Sounds**
* Edit **Play** and **Comment** commands

### Activity 2: Debugging

* Create a program in Code Jumper with incorrect sequencing and fix it in CJ Threads
* Use sequencing to edit sequence values
  + Update the first value to change the sound
  + Update the second value to change the speed or duration
* Add the **Pause** command for effect
* Edit the **Pause** command length
* Be creative! It is an open-ended activity, so the options are limitless.

### Activity 3: CJ The DJ

* Select an appropriate sound set
* View the sounds contained in the sound set
* Add the first sound and select **Sound 1**
* Add subsequent sounds as appropriate
* Add **Pause** commands in appropriate places

## Vocabulary

Learn the key coding terms to help you speak Python fluently in CJ Threads!

In coding, **sequence** refers to the specific order in which steps are followed. Just like brushing your teeth—first you apply toothpaste, then you brush—each action must happen in the correct order to achieve the desired result. **Debugging** is the process of finding and fixing errors in your code, similar to changing dead batteries in a remote control that has stopped working. A **comment** is a note in your code meant for humans to read; it doesn't actually run when the program does. For example, #This plays sound 1 is a comment. Don't worry too much about grammar when writing comments. **Indentation** involves placing blank spaces at the beginning of a line to visually group sections of code, much like starting a new paragraph in a book to separate ideas. A **file extension** identifies the type of file you’re working with—such as .py for a Python file. A **function** is like a mini program within your code; you can think of it like a recipe in a cookbook that performs a specific task. Lastly, a **sound set** is a group of related sounds, similar to organizing your favorite songs into a playlist.

### **Vocabulary Bingo Activity**

**Teachers:** Create a bingo board using the vocabulary words from the lesson. As you read real-life examples that represent each word, students will listen carefully and cover the word they believe matches the example. Feel free to use the examples provided, or come up with your own!

Whoever gets **four in a row** first—across, down, or diagonally—wins!

**Pro Tip:** Leave some blank spaces on the board and add new vocabulary words as students move through the rest of the CJ Threads lessons. Then, to keep the learning fresh and fun, play Vocabulary Bingo again after each new set of words is introduced!

### **Commands**

**sounds = SG.GetSoundGroupWithName("Hello World"):**

The function **GetSoundGroupWithName**is part of the **soundgroup**class. It takes one parameter, which is the name of the sound set to load into memory. Once the sound set has been loaded into the variable **sounds**, it can be passed to the **Play**command to play sounds from the sound set.

**player.Play(sounds, 1, 2):**

The function **Play**will play the appropriate sound in the given sound set at the given pitch. The function takes three parameters.

The **first** parameter is the **sounds**variable, which contains the sounds in the given sound set.

The **second** parameter references the sound in the sound set to play. This value refers to the sound in the sound set from 1 to 8. If a value greater than 8 is attempted, then it will execute the modulus to assign a maximum. Modulus is an operation that keeps the value in the range from 1 to 8. If there are fewer than 8 sounds in the sound set, then sounds will begin repeating.

The **third** parameter refers to the pitch. It accepts values in the range of 1 to 4 and will execute the modulus to assign a maximum value of 4. The following values designate pitch adjustment:

* 1 – Half speed
* 2 – Normal speed
* 3 – One and one-half speed
* 4 – Double speed

For sounds in the midi sound category, the second value refers to the length of the sound with the following values:

* 1 – One quarter second
* 2 – One half second
* 3 – One second
* 4 – Two seconds

**player.Pause(1):**

The function **Pause**will add a small pause before the next command is executed. The first parameter accepts values in the range 1 to 4 and will convert values via modulus into that range. The following values correspond to the following pause lengths:

* 1 – One quarter beat
* 2 – One half beat
* 3 – One beat
* 4 – Two beats

## Getting Set Up for Activity 1, “Hello World!”

Writing your first program called “Hello, World!” is like saying hello to the amazing world of coding! It’s super easy—you tell the computer to say “Hello,” and it does! How cool is that? This small step shows you how to speak the computer’s language and start creating awesome things like games, apps, and more. It’s like unlocking a secret door to endless possibilities!

Guess what? Code Jumper is used by kids just like you all around the world—kids who speak English, German (Germany), Portuguese (Brazil), Spanish (Spain), French (France and Canada), and Spanish (Latin America). And in CJ Threads, you’ll write your very first Python program to say “Hello!” But there’s more—with Code Jumper, you’ll also hear how “hello” sounds in different languages! Get ready to explore Spanish, Afrikaans (South Africa), German, Portuguese, Italian, French Canadian, English, and even Chinese (Mandarin). Are you excited to give it a try? Let’s get coding!

## Item Needed:

* The CJ Threads application

This will open the Code Jumper app as well (the original app)

*Note to reader: You should always launch CJ Threads (the app runner) first because this will launch the Code Jumper app simultaneously.*

## Instructions:

1. Open the application called **CJ Threads** (this will open the Code Jumper app as well) .
2. In the heading, navigate to the **CJ Threads** icon.
3. In the **Code** window, navigate to **Line 016**.
4. Navigate to the **Toolbox** and find the **Edit** button.
5. Select the **Edit** button (**ALT + D**).
6. You’ve opened the **Sound Group** dialogue box. In the drop-down menu under **Sound Category**, change the selection to **CJ the DJ**.
7. In the drop-down menu under Sound Set, **Hello World** is automatically selected.
8. Select and activate the **OK** button.

**Tip:** You might want to write this down or braille it on a sheet of paper so you can easily refer to it while working on your first “Hello, World!” program. In CJ Threads, the app will show only the numbers for the sounds, not their names. Keeping this list nearby will help you know which language each number represents.

Here’s the list of sounds and their corresponding languages:

|  |  |
| --- | --- |
| **Sound** | **Language** |
| Sound 1 | Afrikaans |
| Sound 2 | English |
| Sound 3 | French Canadian |
| Sound 4 | German |
| Sound 5 | Italian |
| Sound 6 | Mandarin |
| Sound 7 | Portuguese |
| Sound 8 | Spanish |

Make sure to have this list handy so you can choose the right sound when creating your first program, Hello World!

## Activity 1: Coding Around the World: “Hello, World!”

Hello, explorers! Did you know that “Hello, World!” isn’t just for programming—it’s also how people greet each other around the world? In this lesson, we’ll create a program in CJ Threads that speaks “Hello, World!” in eight different languages. You’ll use the Play, Pause, and other fun buttons to build a sound adventure.

### CJ Threads Toolbox Buttons you will use:

1. **Comment button:** Use the **Comment** button to add helpful notes or explanations to your code. Comments are like little reminders or messages that only the programmer can see—they won’t affect how your program runs. They’re perfect for explaining what your code does or leaving tips for yourself and others! **Because Python skips these lines, comments can also allow you to isolate a section of code if you suspect that section is causing problems, called *bugs*, in coding. We’ll explore the process of debugging later in this activity.**
2. **Play button:** Plays the selected phrase.
3. **Pause button:** Adds space between phrases (for some users this can make it easier to understand what you just heard).

When working in CJ Threads and Python, your files will end with **.py**. This is called a **file extension**, and it tells us (and the computer) that the file is written in Python. When naming your file, pick something that relates to your project—like “Activity 1: Hello, World in 8 Languages!”—so it’s easy to find later.

**Pro Tip:** Save your work often! Not saving your file is like baking a cake and forgetting to turn on the oven—you’ll lose all your hard work, and no one likes that! So, keep hitting **Save** to keep your progress safe!

### Comment Button

Let’s get started by adding **comments** to our code! We’re going to begin by commenting on Line 17. In every programming language, we can add comments to our code. Comments are special messages that don’t actually affect the program—they’re skipped when the code runs and aren’t visible to the user. Comments help programmers explain what the code does, leave notes for things to change or add, or even ask questions for others who might work on the code later. Let's use comments to make our program clearer and easier to understand!

* In the **Toolbox**, locate the **Comment** button (**Alt + C**) and click to activate it. In the **dialogue box**, type the name of your program. For example, the authors of this lesson named it: **Activity 1: Hello, World in 8 Languages!** Naming your program helps your teacher of students with visual impairments, paraprofessional, parent, or computer science teacher know which activity you’ve created. It’s also a handy way to remember your work so you can easily return to it later and continue coding!

### Notes from a Pro Coder!

**Creating Comments Tip:** Here’s something cool you should know if you want to work as a programmer using Python one day! On Line 017, a **#** symbol (or pound sign) will appear in front of the text you typed as your comment. In Python, the # is used for single-line comments, while three single quotes (''') or three double quotes (""") are used for multi-line comments.

But here’s the best part: CJ Threads does the hard work for you! It automatically adds the right symbols for your comments, so you don’t have to worry about punctuation. Just type your message and focus on creating amazing code—just like a pro!

### Play Button

When writing instructions in Python, everything must follow the rules of the Python programming language. Python has built-in functions, which are sections of code designed to perform specific tasks. One of the most important and commonly used functions is called **print**.

In CJ Threads, the **Play** button (which also represents the physical Play Pod in the Code Jumper kit) acts just like the **print** function in Python. It’s how you display messages or output in your program in CJ Threads.

**First Language (English)**

1. Make sure you are on **Line 017**. This is where you wrote the comment **# Activity 1: Hello World, in 8 Languages!**
2. Navigate to and activate the **Play Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **2** and leave the speed set at **2**.
4. Activate the **OK** button.

The code should appear as follows on Line 018: **player.Play (sounds, 2, 2)**

**Note:** Notice that the command is indented four spaces, so that it is included as part of the Thread1.

1. Activate the **Run** button (**Alt + R**).

#### What did you hear?

You should’ve heard “Hello, World!” in English. Congratulations! You’ve just written “Hello, World!”—your very first program in CJ Threads and Python!

**Fun Fact**: English is always evolving. A new word is added to the English dictionary about every 2 hours. That is an average of 4,000 new words every year.

Now, let’s add a comment to celebrate your achievement! Remember the **Comment** button we introduced earlier? Make sure you’re on **Line 018**, activate the **Comment** button (**Alt + C**), and type the following (or create your own phrase):

**Hello World! English**

**Next Language (Afrikaans)**

1. Make sure your cursor is on **Line 19**, directly after or before the comment you wrote.
2. Navigate to and activate the **Play** **Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **1** and leave the speed set at **2**.
4. Activate the **OK** button.
5. Activate the **Run** button (**Alt + R**).

#### What did you hear?

You should’ve heard “Hello, World!” in Afrikaans! To say “Hello, World!” in Afrikaans, you would say: **“Hallo, Wêreld!”**

**Fun Fact:** Afrikaans is one of the youngest languages in the world, spoken mainly in South Africa and Namibia. It grew from Dutch settlers in the 1600s and has about 7 million speakers today! Isn’t it amazing how far this “young” language has come?

#### Next Step: Add a Comment!

1. Make sure your cursor is on **Line 020**, directly before or after the **Play** function you just created.
2. Navigate to and activate the **Comment** button (**Alt + C**).
3. Type, “**Hello, World! Afrikaans!**“ into the edit field (or see below to learn how to type it in Afrikaans!). If you are using a laptop, you may not have a numeric keypad with a **Num Lock** key. Therefore, you will not be able to type it in Afrikaans.

### Notes from a Pro Coder!

If you want to write the comment in Afrikaans, you can type:

**#Hallo, Wêreld! Afrikaans!**

To type the accent mark used in **#Hallo, Wêreld! Afrikaans** (**ê**, called an **e-circumflex,** or, an E wearing a party hat!) on a Windows PC, follow these steps using the Alt Code method:

1. Make sure the **Num Lock** key is turned on.
2. Hold down the **Alt** key.
3. While holding the **Alt** key, type **0234** on the numeric keypad.
4. Release the **Alt** key, and an **ê** will appear!

This alt code is handy for adding special characters like the circumflex accent used in “Wêreld”!

**Next Language (French Canadian)**

1. Make sure your cursor is on **Line 21**, directly after or before the comment you wrote.
2. Navigate to and activate the **Play** **Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **3** and leave the speed set at **2**.
4. Activate the **OK** button.
5. Activate the **Run** button (**Alt + R**).

**What did you hear?**

You should’ve heard “Hello, World!” in French Canadian!

To say “Hello, World!” in French Canadian, you would write: **Bonjour, le Monde!**

**Fun Fact:** French Canadian, also known as Québécois, is a variety of French spoken mainly in Quebec, Canada. It has unique words and expressions that make it different from European French. For example, they often say “Salut!” for “Hi!” in casual conversations!

#### Next Step: Add a Comment!

1. Make sure your cursor is on **Line 022**, directly before or after the **Play** function you just created.
2. Navigate and Activate the **Comment** button (**Alt + C**).
3. Type, “**Hello, World! French Canadian!”** into the edit field or you could type, “**Bonjour, le Monde! French Canadian!”.**

**Next Language (German)**

1. Make sure your cursor is on **Line 23**, directly after or before the comment you wrote.
2. Navigate and activate the **Play** **Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **4** and leave the speed set at **2**.
4. Activate the **OK** button.
5. Activate the **Run** button (**Alt + R**).

**What did you hear?**

You should’ve heard “Hello, World!” in German!

To say “Hello, World!” in German, you would write: **Hallo, Welt!**

**Fun Fact:** German is the most widely spoken native language in Europe, with over 90 million speakers! It’s also famous for having super long words, like “Donaudampfschiffahrtselektrizitätenhauptbetriebswerkbauunterbeamtengesellschaft”—a word that’s over 80 letters long!

#### Next Step: Add a Comment!

1. Make sure your cursor is on **Line 024**, directly before or after the **Play** function you just created.
2. Navigate and Activate the **Comment** (**Alt + C**) button.
3. Type, “**Hello, World! German!”** into the edit field or you could type, “**Hallo, Welt! German!**

**Next Language (Italian)**

1. Make sure your cursor is on **Line 25**, directly after or before the comment you wrote.
2. Navigate and activate the **Play Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **5** and leave the speed set at **2**.
4. Activate the **OK** button.
5. Activate the **Run** button (**Alt + R**).

**What did you hear?**

You should’ve heard “Hello, World!” in Italian!

To say “Hello, World!” in Italian, you would write: **Ciao, Mondo!**

**Fun Fact:** The Italian alphabet, which derives from the Latin alphabet, has only 21 letters. The letters j, k, w, x and y are not included in native words.

#### Next Step: Add a Comment!

1. Make sure your cursor is on **Line 026**, directly before or after the **Play** function you just created.
2. Navigate to and activate the **Comment** (**Alt + C**) button.
3. Type, “**Hello, World! Italian!”** into the edit field or you could type, “**Ciao, Mondo! Italian!”.**

**Next Language (Mandarin)**

1. Make sure your cursor is on Line 27, directly after or before the comment you wrote
2. Navigate and activate the **Play Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **6** and leave the speed set at **2**.
4. Activate the **OK** button.
5. Activate the **Run** button (**Alt + R**).

**What did you hear?**

You should’ve heard “Hello, World!” in Mandarin!

To say “Hello, World!” in Mandarin, you would write: **Nǐ hǎo, shì jiè!**

**Fun Fact:** The Mandarin language used to be written in vertical columns going from right to left. Since 1955, it has taken a more westernized approach and is written horizontally from left to right.

#### Next Step: Add a Comment!

1. Make sure your cursor is on **Line 028**, directly before or after the **Play** function you just created.
2. Navigate to and activate the **Comment** (**Alt + C**) button.
3. Type, “**Hello, World! Mandarin!”** into the edit field or you could type “**Nǐ hǎo, shì jiè! Mandarin!”.**

**Next Language (Portuguese)**

1. Make sure your cursor is on **Line 29**, directly after or before the comment you wrote.
2. Navigate to and activate the **Play Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **4** and leave the speed set at **2**.
4. Activate the **OK** button.
5. Activate the **Run** button (**Alt + R**).

**What did you hear?**

You should’ve heard “Hello, World!” in Portuguese!

To say “Hello, World!” in Portuguese, you would write: **Olá, Mundo!**

**Fun Fact:** There are between 500,000 and 800,000 words in Portuguese. This could possibly mean Portuguese is one of the richest languages on the planet, vocabulary-wise. The majority of speakers only know about 1,000 to 5,000 words from that large amount.

#### Next Step: Add a Comment!

1. Make sure your cursor is on **Line 030**, directly before or after the **Play** function you just created.
2. Navigate to and activate the **Comment** (**Alt + C**) button.
3. Type, “**Hello, World! Portuguese!”** into the edit field or you could type, “**Olá, Mundo! Portuguese!”.**

**Next Language (Spanish)**

1. Make sure your cursor is on Line 31, directly after or before the comment you wrote.
2. Navigate to and activate the **Play Pod** (**Alt + P**) in the **Toolbox**.
3. Change the sound to **4** and leave the speed set at **2**.
4. Activate the **OK** button.
5. Activate the **Run** button (**Alt + R**).

**What did you hear?**

You should’ve heard “Hello, World!” in Spanish!

To say “Hello, World!” in Spanish, you would write: **¡Hola, mundo!**

**Fun Fact:** There are 30,500 Spanish words that contain all five vowels: **a, e, i, o, and u**. A few examples include **murciélago** (bat—the animal, not the baseball kind!), **aeropuerto** (airport), and **educativo** (educational).

#### Next Step: Add a Comment!

1. Make sure your cursor is on **Line 032**, directly before or after the **Play** function you just created.
2. Navigate to and activate the **Comment** (**Alt + C**) button.
3. Type, “**Hello, World! Spanish!”** into the edit field or you could type, “**Hola, Mundo! Spanish!".**

## **Answer Key for Hello World**

**014:def Thread1():**  
**015:    global x**  
**016:    sounds = SG.GetSoundGroupWithName("Hello World")**  
**017:    # Activity 8:  Hello, World in 8 Languages!**  
**018:    player.Play(sounds, 2, 2)**  
**019:    # Hello World! English**  
**020:    player.Play(sounds, 1, 2)**  
**021:    # Hallo Wêreld! Afrikaans**  
**022:    player.Play(sounds, 3, 2)**  
**023:    # Bonjour, le monde! French Canadian**  
**024:    player.Play(sounds, 4, 2)**  
**025:    # Hallo Welt! German**  
**026:    player.Play(sounds, 5, 2)**  
**027:    # Ciao, mondo! Italian**  
**028:    player.Play(sounds, 6, 2)**  
**029:    # Nǐ hǎo, shìjiè! Mandarin**  
**030:    player.Play(sounds, 7, 2)**  
**031:    # Olá, mundo! Portuguese**  
**032:    player.Play(sounds, 8, 2)**  
**033:    # ¡Hola, mundo! Spanish**

Amazing work, coders! Now you’ve learned how to say “Hello, World!” in eight languages and even created your own multi-language sound code. Which greeting was your favorite? Remember, programming is a universal language—just like saying “Hello!”

## **Activity 2: Debugging**

1. Make sure the hub is powered on and connected to Bluetooth on your PC.
2. Connect four Play Pods to **Thread 1** on the hub.
3. On the physical side of Code Jumper, set Pod 1 to **Twinkle 2**, Pod 2 to **Twinkle 1**, Pod 3 to **Star**, and Pod 4 to **Little**. For now, set all speeds to **1 times speed**. You may notice that in CJ Threads, normal speed has a value of **2**.
4. Observe the code in both the Code Jumper block coding language window and in the CJ Threads Python code window. Make note of similarities and differences.
5. Play the code by pressing the play button or using the **Play** hot key (Alt + P).
6. Jot down in your computer science journal or notetaking program what song this should be and how the song is broken currently. Create a plan to fix this code.
7. Use your plan to guide your investigation to fix the code.
8. Answer the questions below:
   1. What lines did you have to edit (Alt + D) on the Python side to fix the song?
   2. Were there any lines that stayed the same between running the broken code and running the fixed code?
   3. How does this activity demonstrate the importance of sequencing? Was the original code following an expected sequence? If not, how did you know? What steps did you take to fix the sequencing to what you were expecting?
9. Check for understanding! To really prove your worth as a galactic DJ, pair off into groups of two and put your friend to the ultimate test. One of you will play the “programmer” first and the other will be the “tester.” Then, you will swap roles and repeat the same process.
10. Once you have decided who will be the programmer and tester this round, the programmer should write their own broken version of “Twinkle, Twinkle Little Star.”
11. Once you are satisfied that you have reached maximum bamboozlement, pass your code to the tester.
12. Ask the tester to fix your code. To up the ante and make this a competition, you can time each other.
13. Switch roles and repeat Steps 9 through 12.

\*Note: To keep the programmer’s code secret, testers might need to wait on the other side of the classroom, outside the classroom, or facing away from the computer.

|  |  |
| --- | --- |
| **Code Jumper Code** | **CJ Threads Code** |
| Thread 1 Twinkle, Twinkle | 016: sounds = SG.GetSoundGroupWithName("Twinkle, Twinkle") |
| Play Twinkle 2 for 1 times speed | 017: player.Play(sounds, 2, 2) |
| Play Twinkle 1 for 1 times speed | 018: player.Play(sounds, 5, 2) |
| Play Star for 1 times speed | 019: player.Play(sounds, 4, 6) |
| Play Little for 1 times speed | 020: player.Play(sounds, 7, 6) |

## **Activity 3: CJ The DJ**

**Characters you’ll meet in this activity:**  
 Get ready to team up with some out-of-this-world companions from the planet **Jumpiter**! You’ll join **CJ**, a curious and energetic cat-like alien who loves exploring new sounds and solving creative challenges. Along for the ride is **Melody**, CJ’s best friend—a clever, calm, and rhythmic fellow cat-like alien who always thinks one step ahead. Helping guide the journey is the quirky and wise **Professor Loopstein**, a slightly eccentric cat-like alien who’s full of knowledge, loves to teach, and isn’t afraid to show off his funky dance moves.

#### **CJ The DJ**

#### CJ has been jumping for joy ever since he got accepted into the **Computer Science and DJ program** at **Byte Beats Academy** on **Planet Jumpiter**! He’s double majoring in **computer science and music**, and his very first assignment in class was to write the classic “Hello, World!” program.

But now, things are getting more exciting—CJ gets to **create his very own beat**, one that’ll be used throughout the rest of the semester. His teacher challenged the class to think carefully about the kind of music they want to make. So, CJ decides to create a beat that’s **perfect for an epic end-of-class party** *(the teacher hasn’t told CJ or his classmates yet, but they’ll actually be competing in a song battle—and the winner will get to record their track made in the CJ Threads Program and share it with a representative from APH! The winning song will be played for the world on APH’s podcast and featured on their social media!)*

*(Also: Did you know APH stands for* ***Awesome Planet Hits****, the coolest kid-run station in the galaxy?! Now that you do, you’re out-of-this-world cool too!)*

Even though CJ’s super excited to use the **CJ Threads program**—the special tool their teacher is having them use—the class is reminded:

“Start with the end in mind. We’re going to take it slow and build something awesome, step by step.”

CJ and his classmates can’t wait to get started! Their teacher handed out the step-by-step directions, and everyone sat down at their workstations, ready to jam out using the newest version of CJ Threads—the super cool music and coding program they will be learning at Byte Beats Academy!

Follow the steps below just like CJ and his friends—and when your teacher lets you know it’s time to switch classes or the bell rings, be sure to save your program so you’ll have it ready for the next session at Byte Beats Academy on Planet Jumpiter!

### **Name Your Program**

Right before CJ and his classmates started their independent work, their instructor, Professor Loopstein, called out:

**“One more thing before you begin! It’s *super* important to name your program and save it!”**

For this lesson, CJ and his classmates are making a **drum track** or **beat track**. In music and production, this is called the **rhythm track** or **foundation track**. It’s the part that sets the **tempo**, **groove**, and overall **feel** of the song.

🎧 In styles like **pop**, **hip-hop**, or **electronic music**, people often just call it **“the beat.”**

Since CJ is training to be a DJ, his professor encourages him to name it using the term:

***The Beat.***

### **Steps to Name Your Program:**

1. Make sure you’re on **Line 001**.
2. Press **Alt + D** to open the **Edit** box.
3. In the **Edit box**, type: **The Beat**.
4. **Tab** to the **OK** button and press **Enter**.

When you’re done, **Line 001** should read as **001: # The Beat**

### **To Save Your Program:**

1. Press **Alt + S** to open the **Save** dialog box.
2. In the box that appears, type the name: **The Beat**.
3. Press the **Tab** key to move to the **OK** button, then press **Enter**.

Your program is now saved!

Remember from earlier: Your program will be saved with a **.py** file extension.  
 That just means it’s a **Python** file—because CJ Threads uses the Python programming language!

**To Change the Sound Set**

1. Use your mouse or arrow keys to go to **Line 016**:

**sounds = SG.GetSoundGRoupWithName("Hello World")**

1. Press the key command **Alt + D** to open the **Edit box**.
2. You’ve opened the **Sound Group dialogue box**. In the drop-down menu **under Sound Category**, change the selection to **CJ the DJ**.
3. In the drop-down menu under **Sound Set**, change it to **Rhythm**.

### **Try All Eight Sounds!**

1. Make sure your **cursor** is on **Line 016**.
2. Use the **Play Pod** command by pressing **Alt + P** to open the **dialogue box**.
3. For the first sound, leave it set to **Sound 1**.
4. Navigate to the **OK** button by pressing **Tab** until you reach it, then press **Enter**,  
    or click **OK** with your mouse.
5. To hear the **code you’ve just “written” on Line 017**, press **Alt + R** to **Run** your program.
6. Make sure you’re on **Line 017**: **player.Play(sounds, 1, 2)**
7. Press the **Edit** command **(Alt + D)**.
8. Change the **Sound** number to **2** by typing in the number **2**.
9. Navigate to the **OK** button (**Tab → OK**), then press **Enter**.

Repeat Steps **6 through 9** for **all eight sounds** in the set by changing the **Sound** number each time—from 1 to 8.

**Tip:** If you’d like, you can **write down what each sound reminds you of** using:

* A sheet of **paper**
* A **braille writer** or **braille notetaker**
* A **Word document**
* A **recording app** on your phone or tablet

This will help you remember which sounds you liked best and what they sounded like to *you*—so you can pick the perfect ones as you build your beat!

**Time to Drop the Beat Like a Real DJ!**

Before we dive in, make sure your **Sound Set** is set to **CJ The DJ** on **Line 016**.  
 Didn’t do that yet? No worries—scroll up to the section titled **To Change the Sound Set** and follow those steps first

**Let’s Build Your Beat!**

If you created a list of sounds earlier, have that ready to reference—it’ll make this next part easier!

* Make sure your **cursor is on Line 016**.
* Next, locate the **Play** button in the **Toolbox**. Activate it using the key command:  
   **Alt + P**
* Once the Play Pod dialog box opens, change the sound to the one you want to kick off your beat. You’re officially making music!

### **CJ, Melody Harmonia & the Rhythm Track**

Professor Loopstein is showing CJ and his beat buddy **Melody Harmonia**  how to build a rhythm track using CJ Threads. Follow along so you can create your own groove too!

CJ and Melody are both fans of **Sound 2**, and they want to start their beat with four Play Pods using that sound.

#### **Here’s How:**

1. Place your cursor on **Line 016**.
2. Press **Alt + P** to open the Play Pod dialog box.
3. Type **2** for **Sound 2**.
4. Press **Tab** to navigate to **OK**, then press **Enter** (or click **OK** with your mouse).
5. **Line 17** will have your first rhythm Play Pod sound**.**

Repeat these steps on:

* **Line 018**
* **Line 019**
* **Line 020**

Now you’ve got four solid Play Pods pumping out Sound 2. Nice work!

### **Add Some Groove with Pauses (AKA “Rests”)**

After listening to their track, CJ and Melody decide to add some space between the sounds—like in their favorite APH Radio jam, “Moonwalk Funk.”

Professor Loopstein, thrilled, starts humming and doing a funky moonwalk while clapping his paws. CJ and Melody just laugh and throw their paws in the air!

He explains how to add **Pauses** (also called **Rests**) to create space between sounds using CJ Threads.

#### **Follow these steps to add Pause Pods (rests) after each Play Pod:**

1. Navigate to **Line 017**
   1. Press **Alt + U** to open the Pause Pod
   2. Type **2** (for a two-beat pause)
   3. Tab to **OK**, then press **Enter**
   4. Now **Line 018** should have your first Pause Pod sound.
2. Repeat the above steps so the Pause Pods are on the following lines:
   * **Line 020**
   * **Line 022**
   * **Line 024**

(Each of the pause pods should come *after* a Play Pod.)

### **Ready to Rock?**

Now go to **Run** and hit **Play** to hear your custom rhythm track in action!

You just built a beat *and* added musical pauses like a professional DJ. CJ and Melody would be proud!

### **Bonus Challenge: The Remix Zone**

Ready to take your beat to the next level?

Here’s your **Extra Challenge Activity**:

**Switch It Up With Different Sounds and Patterns!**

1. Try replacing **Sound 2** with a new sound (like 3 or 5).
2. Create a pattern, for example: **Sound 2, Pause, Sound 3, Pause, Sound 2, Pause, Sound 5**.
3. Use different **pause durations** (try 1, 3, or even 4) to explore rhythm variation.
4. Add **more Play Pods** on Lines 025 to 030 to extend your track.

When you’re done, hit **Run** and share your custom remix with a classmate!

At the end of class, Professor Loopstein turns to CJ, Melody, and you with a big grin and says:

“Bravo, beat builders! CJ, Melody, and you—yes *you*—brought the beats and the brains! You rocked the code today and made your first rhythm shine—I’m officially impressed! Next time, we’re turning up the coding dial as we jump into **constants**, **randoms**, and **variables**—oh yes, it’s a *code party* waiting to happen! Whether your beats are locked in, totally unpredictable, or somewhere in between, these coding tools will help you mix it all together. Until then, keep that curiosity groovin’ and those code fingers movin’!”