BRAINY BYTES®

PLUG INTO CREATIVITY

GAME LAB

CODE CREATE PLAY

INSTRUCTOR PLAYBOOK

Sample Excerpt

A BRIANY BYTES PATHWAY CURRICULUM

WELCOME TO BRAINY BYTES CODING PATHWAYS

Welcome to your Brainy Bytes adventure! We are so excited for you to bring this curriculum to life and guide your students through a fun, creative, and confidence-building experience. We know you and your students are going to love exploring, creating, problem-solving, and celebrating what they build along the way.

PURPOSE

EMPOWER • ENRICH • INSPIRE

After school and homeschool classes are the core of Brainy Bytes' school year. They should serve to stimulate the child's interest and encourage them to delve deeper into STEAM.

Keep in mind that our programs come after a full day of school for your student. Your goal should be to provide a fun, nurturing and positive environment that fosters innovation, problem-solving and collaboration. Build relationships with your students and encourage them to do the same with their classmates. Do this, and you will instill a love of learning and keep families coming back for more for years to come.

OBJECTIVES

IGNITE A SPARK FOR LEARNING

It is our policy at Brainy Bytes to make sure that each student is learning. We strive to help teachers manage their classrooms with clear expectations and active student participation. This will ensure that students leave our courses with a new found skill and, hopefully, a passion for learning even more.

This involves moving from instruction-driven class time to a question/participation-driven class time. As you progress, the curriculum will turn to a more student-led format with questions and activities that encourage them to use what they have previously learned.

Game Lab utilizes video game creation to teach basic programming terminology and skills. Students will learn how to create multiple video games using Scratch. Through this project-driven lesson plan, participants will not only be introduced to fundamental coding steps; they will learn the analytical skills needed to master many programming tasks successfully, all while making their own game to share and build on even after the course.

BEFORE YOU START

Read the entire curriculum! Make sure that you do everything in the curriculum yourself. Never teach anything that you haven't done yourself.

Familiarize yourself with the software! Know and become comfortable with the key programming steps used throughout the curriculum and beyond.

Make sure you have successfully created all 3 of the games that are taught. Be prepared to move beyond the curriculum as we encourage the students.

Equipment Needed

- Student laptop (1 per student)
- 1 Teacher laptop
- Projector
- Game samples
- Class files & images
- "Recipe for a Good Game" walkthrough
- "Out of the box" activity supplies: 2 small boxes with lids, 1 ball that fits in boxes

GAME LAB COURSE MAP

Week #	Semester Week	Supply Needs & Notes
1	Intro to Programming & Using the platform	
2	Creating a Top-Down Shooter Game *Scratch, Environment*	
3	Creating a Top-Down Shooter Game *Player*	
4	Creating a Top-Down Shooter Game *Projectile, Player*	
5	Creating a Top-Down Shooter Game *Enemy, enemy lives*	
6	Creating a Top-Down Shooter Game *Player lives, end screen*	
7	Creating a Top-Down Shooter Game *Portal, levels*	
8	Creating a Top-Down Shooter Game *Boss*	
9	Creating a Top-Down Shooter Game *End Portal, Extras*	
10	Creating an Infinite Runner *Beginning*	After omitting free building, Infinite Runner could be omitted to provide shorter course or make up time
11	Creating an Infinite Runner *Finish*	
12 & 13	Free Building	Can be omitted if you need to provide shorter course or make up time
14	Game Day	Collect games to send home.

HOW TO TEACH FROM THIS PLAYBOOK

This curriculum is designed to help you teach with confidence, consistency, and energy. It is not meant to be read silently by students or used as a loose activity guide. The lessons are written to help you lead the class, guide discussion, demonstrate each step, and support students as they build.

As you move through each lesson, pay close attention to the formatting cues. These cues tell you what to say, what to do, what to emphasize, and when to adjust for your students' pace.

CIRRCULUM LANGUAGE GUIDE

When you see...	What it means
Blue italic text	Instructor-only direction, reminders, pacing notes, or teaching guidance
Regular black text	Content to share, explain, or discuss with students
Red text	Key terms, important reminders, or course-adjustment notes
Bold text	Programming steps, software actions, or important build instructions
Helpful Hint boxes 	Classroom-tested tips for instruction, student engagement, software use, troubleshooting, or pacing

HOW TO USE THE LESSON PAGES

Read through the full lesson before class and complete the project yourself before teaching it. During class, use the student-facing language as your guide, but speak naturally as you demonstrate each step. For example, when the curriculum says click on ____, you might say, "Now we are going to click ____."

Do not skip the discussion, review questions, or explanation sections. These are included to help students understand what they are building, not just follow directions. The goal is for students to learn coding concepts, practice problem-solving, and gain confidence as creators.

TEACHING PACE

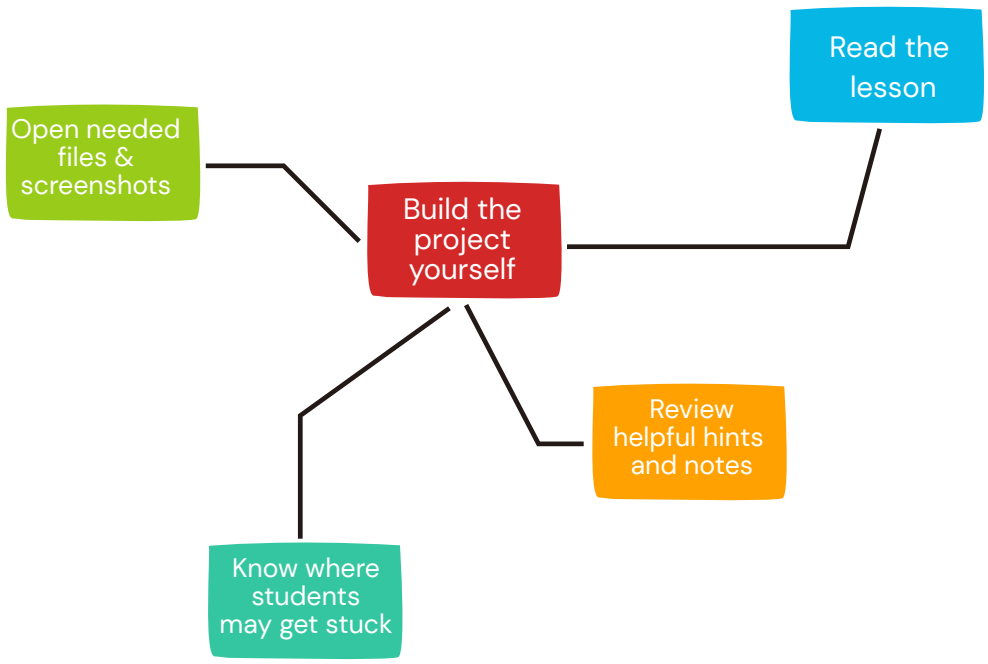
This curriculum is organized around the session breakdown, but every class will move at a slightly different pace. Some groups may need extra review, while others may be ready for more challenge. Adjust when needed, but remember: the goal is to teach skills, not rush students through the motions.

When students need more time, slow down, review, troubleshoot, and reinforce the concept before moving on. When students finish early, use extension challenges, extra testing time, or project-sharing moments to deepen learning.

INSTRUCTOR MINDSET

Your role is to lead, encourage, question, demonstrate, and support. Help students think through mistakes instead of simply fixing problems for them. Ask questions, celebrate progress, and create a classroom environment where students feel comfortable experimenting, debugging, and improving their work.

PREPARE BEFORE YOU TEACH



GLOSSARY

Familiarize yourself with the key programming terms you will need to know to teach coding.

Code The step-by-step instructions that programmers create and use to tell a computer what to do.

Programming Inputting code into the computer via different computer languages.

Events Something a computer program can react to, such as a key being pressed, or the mouse being clicked.

Conditions/Conditional A "true/false" statement used to make a decision. Statements that only run under certain conditions.

Variables A place to store data that can change in a program, such as the player's score.

If Then Statements/If Then Else Statements Common programming structure that implements "conditional statements", such as IF enemies = 0 THEN end game.

For Loop A loop with a predetermined beginning, end, and increment (step interval).

While Loop A loop that continues to repeat while a condition is true.

Pixel Short for "picture element", the colored dots on a screen that make up graphics.

Sprite (as defined by Scratch) A picture on the stage in Scratch that code blocks can move and change.

X Coordinates & Y Coordinates Coordinates are used to pinpoint a location on the stage/screen. Just like graph coordinates, x numbers are for horizontal position and y numbers are for vertical.

Clone (Scratch) An identical copy of a sprite.

LESSON 2
SCRATCH ENVIRONMENT

DAILY INTRO & REVIEW

Before the course begins, ensure you have any supplies you may need, and that the projector is set up (if needed).

Let's go over what you have learned so far. Start each day with a quick review of what they learned the day before.

Review with them some of the things they learned last week by asking students if they can tell you the following steps or explanations. Do not skip this step, it is essential to help them retain everything they are going to learn. Go over each bullet and make sure to explain key factors completely.

~ Share 3 things that make a great game.
~ What is coding? Programming?
~ Why is sequence important?
~ What are blocks in scratch?
~ Where do we find blocks on the screen?
~ How do we start/stop our games?

LESSON: INTRO TO TOP DOWN SHOOTER

Ask the students if they have ever played shooter games. Then ask what type of shooter games they play. Be sure to talk about first person shooter games vs third person. Shooter games have been, and always will be, a staple in the gaming industry. If you look back at old-school games from Atari and the original Nintendo, you will see that many of the games fell into the shooter genre. Usually, shooter games include a story of some type and an enemy to defeat. They are excellent at testing a player's reaction time and skill level. It's a perfect pick for the first game we will create.

We are going to make a zombie Top-Down Shooter game. Does anyone know what that is? *Wait for responses. The students should be able to guess if they don't know already.*

A top-down shooter is a shooter game where the levels are controlled from an overhead view (top-down). We see all the characters in the game as if we are looking at them from above.

Open the completed shooter game from your jump drive. Show it in action to the class so they can see what they will be making

This is the game we are going to make together. Are you ready? Let's do this!

HELPFUL HINT

Make sure you are familiar with where your resource folder is so that you can easily access all videos and pictures. We recommend you bring up any files needed for the day before starting each class.

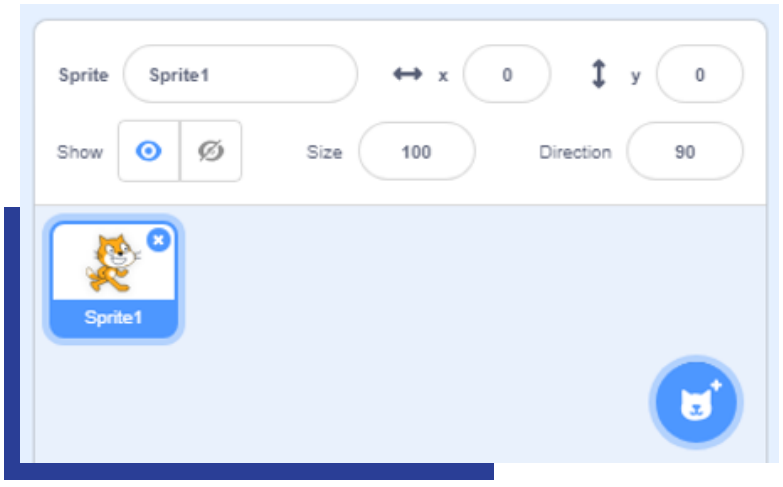
Also, be aware of file locations on the student laptops so you can easily direct them to the files needed.

LEVELS (ENVIRONMENT)

Have the students open Scratch.

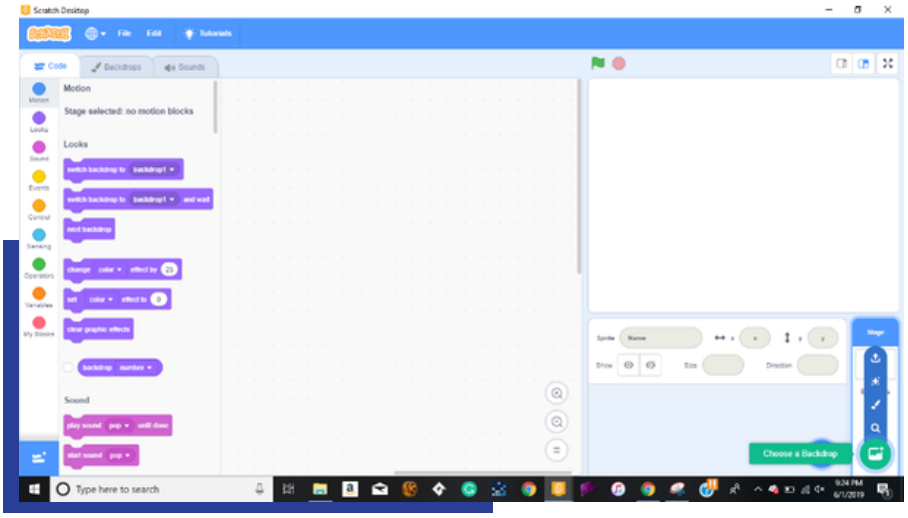
Here, it will depend on what version of scratch you are using to teach the class. If you are using the downloaded version, have the students double click the Scratch icon. If you are using the online version, open the web browser of the location's choice and go to: scratch.mit.edu and then click on create from the top menu – refer to last week's lesson if needed.

Once you are on the programming screen, Scratch the cat will be in your sprite list. This time we want to use a different sprite so let's delete him. Click the X in the top of the sprite1 box with scratch in it. We will add our own sprite in a moment.



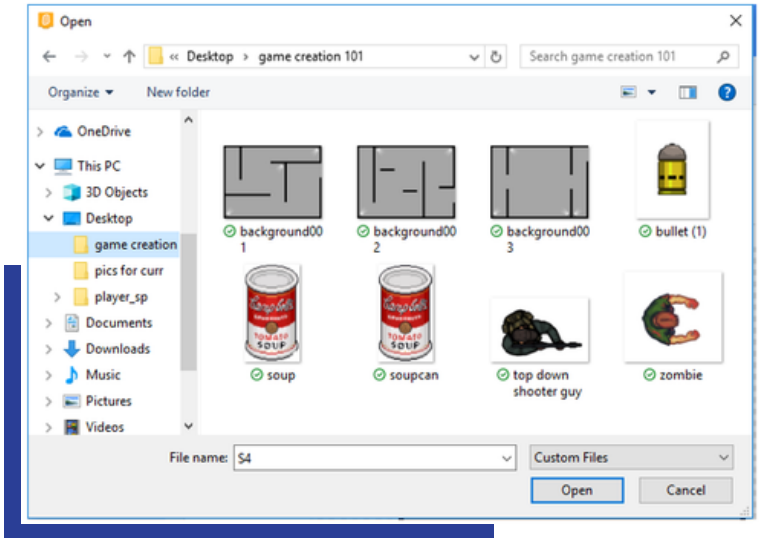
First, let's set the environment for our game. We can't put a player into the game without someplace for them to walk around. Remember, last week, we talked about how we need an environment that makes sense to the story.

Look in the bottom right corner of the screen and you will see the **STAGE** section of the **BACKGROUNDS**. Hover over the background picture and a menu will pop up.



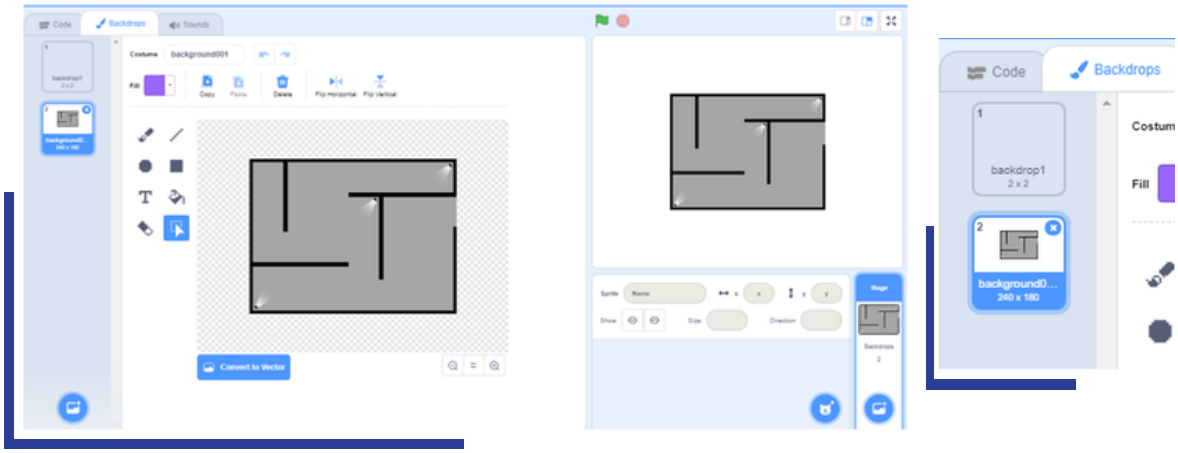
Go up to the top option, which is **UPLOAD BACKDROP**, and click it. 

Brainy Bytes has provided the backgrounds that we need to make our zombie shooter game. Open the **Game Lab** folder and find the **background 1** picture.



Either **double click** the picture or click on it and then click open. Your background will now show in your stage. It will also show under the backdrops tab in the upper right corner. Click on the **Backdrops Tab** to see the background.

Notice that our picture is now the 2nd background in the list on the left side of the screen. We can delete the BLANK background by clicking the **X** in the top right corner of that box.

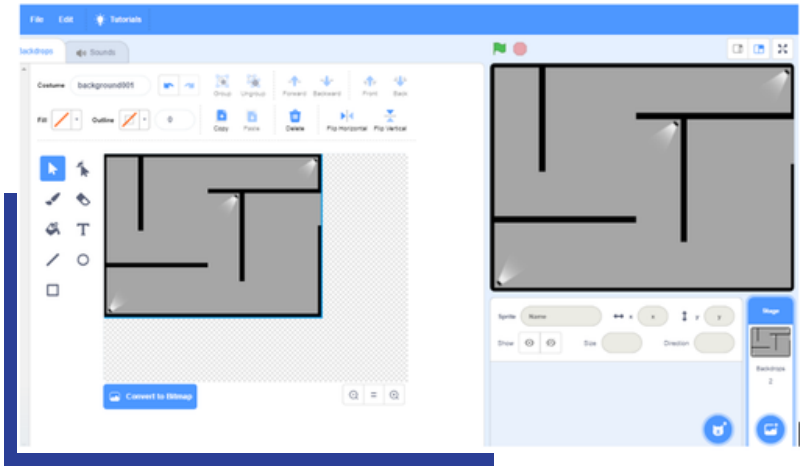


At the bottom of the background picture, click on the large blue button that says **CONVERT TO VECTOR.**

The screen now has additional editing options. Let's improve the placement of our background image so that it fills up the entire stage.

Click on the arrow (**SELECT**). Then, zoom out all the way by clicking on the magnifying glass with the **minus** (**-**) in it. This will give us true to size fit.

Now click anywhere on the picture. This will highlight the border of the picture. **Click and drag the corners** of the picture to fill the entire window.



We want to RENAME the background. Click on the field next to the word **COSTUME** on the screen. We are naming this first environment "**0**".

We will need more costumes than this to add levels. But, for today we just want this one set up with the name "0".

We have our first level's background set.

WATCH FOR

Before moving to the next step, check that each student has completed the background setup correctly and knows where their project files are saved.

PROJECTILE (AMMO)

If there is still time in class, have the students choose an ammo sprite from the folder as described below. Otherwise, this will be covered in week 4.

We are almost done with today's programming. Since we are creating a shooter game, we want to go ahead and create a projectile to program. This is what our player will be shooting at the enemy.

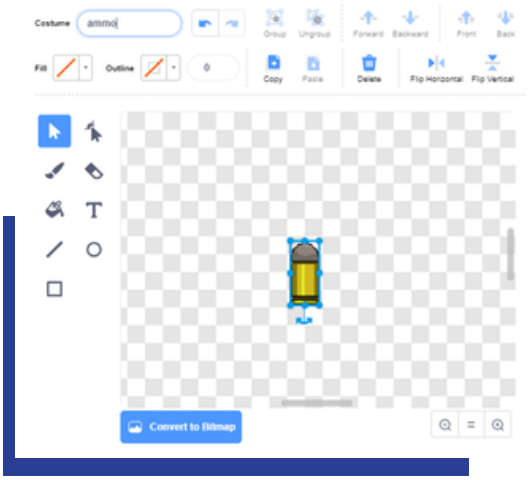
To do this we need to add a sprite. We will upload it from our Game Lab folder.

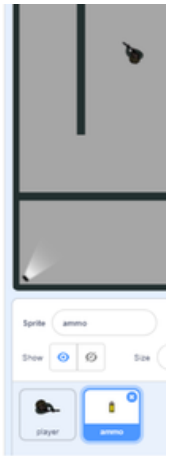
Just like we hovered over the picture to upload a background, we want to hover over the cat in the bottom right corner and choose the top option **UPLOAD SPRITE**.

Navigate to the **Game Lab** folder. Choose the **projectile** that you want your player to shoot and click **open**.

The projectile will be too big when it shows up on the stage. Click on the **costumes tab** which is now in the place of the backdrops tab because we have our sprite selected, and very much like we did in the background let's fix that.

We have to click on the blue **CONVERT TO VECTOR**, zoom out, click select, click on the **sprite**, re-size it by dragging the corners. Set this sprite to around **13 x 6.**





If you are using a sprite that definitely has a forward point to look like it's traveling in a specific direction, you will need to make sure that your player sprite and ammo sprite are pointing in the same direction. If not, the ammo will be off when it is animated to move from the player's weapon. You may have to grab the blue curved arrows and drag it to spin it the correct direction once you add your player.

We cannot be finished until we **CENTER** the sprite. To do this we need to move the projectile to where it is centered **over the crosshairs** in the middle of the screen.

We need to change the name of the **COSTUME** and the name of the **SPRITE**. Let's change the names to **AMMO** with no caps in both fields.

Now we are set to code the ammo.

Have the students look at their sprite on the stage. Notice that the ammo is just sort of sitting there. It would do that even if we had programming to start the game. Do we want our ammo to be there at the start of the game? No. We don't want to see the ammo on the screen until our player actually shoots.

Let's hide the ammo until we need it. But how do we do that? When do we want it to hide? To start a game in Scratch we use the Green Flag. So, when we click the green flag, the game starts. *Try to lead the students to the fact that it needs to be hidden at the start of the game...so when the green flag clicked Let them know that this will hide it till we program it to show up.*

Click the **CODE** tab of the ammo sprite and go to the **EVENTS** section and drag out a **WHEN GREEN FLAG CLICKED.**

We then need to go to the **LOOKS** section. We haven't been here yet. This section has blocks that set up how and what we can see in the game.

Look for and drag out the **HIDE** block. This will hide the ammo until we want it to show up. Notice we don't put this one inside the **FOREVER LOOP**. Why? *Give the students a chance to come up with the answer.*

If the **HIDE** was inside a **FOREVER LOOP** – would we ever see the ammo? No. And we want to see the ammo every time we shoot.

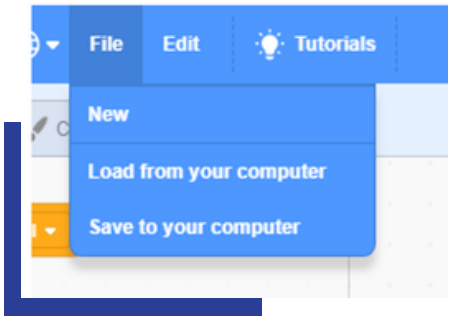
We are going to stop there today. We have set up a lot in our game. Next week, we'll be ready to get into the programming of our first game.

Before you move on to saving, take a moment to make sure everyone has their game where it should be.

SAVE YOUR GAME

Let's **SAVE** our game.

Click on **FILE>SAVE TO YOUR COMPUTER**. This is true for both the downloaded version and the online version.



Call your game **YOUR NAME GAME 1**. Then, you will be able to find it and not have to think about which game is which later.

When you click save to your computer, a window should pop up and ask where you want to save. In this window be sure to **NAME** the game in the field where it says scratch project. If this window doesn't show up and it automatically downloads, I will need to find it and rename it.