

Physical Computing With Micro:bits

Rucha Gokhale

August 3, 2026 · NJ CSPDWeek · The College of New Jersey

Hello! I'm Rucha Gokhale.

- Adjunct Robotics & Python Instructor @Hudson Montessori School
- 20+ years in technology
- 10+ years building with applied AI
- MEng, MBA, BS Mechanical Engineering

My superpower: Bringing technology and people closer together.

What brings you here today?



Let's Define the Big Ideas

Before we begin, let's make sure we share a clear understanding of the key ideas we'll use throughout the workshop.

Physical Computing

Micro:bits

Python Foundations

What Is Physical Computing to You?

ISTE Definition

Using code with devices like sensors, motors, lights, and microcontrollers to connect computing to the real world.

CSTA-Aligned Definition

Working with physical components and devices, including integrating sensors and actuators with computing systems.

Discussion: where do these definitions agree, and where might your own working definition differ?

Why Does It Matter?

In my own teaching, I've seen that many students engage more deeply when code produces something they can see, hear, or interact with. Physical computing gives abstract ideas a real-world outcome, and research supports what I've observed in the classroom.



Why Physical Computing Matters

Physical computing helps students learn computer science by making ideas visible, interactive, and connected to the real world.

- 1 Makes abstract ideas concrete: students see how code, inputs, and outputs work together in real time.
- 2 Boosts engagement and motivation: students build meaningful, tangible projects rather than only screen-based tasks.
- 3 Builds problem-solving and creativity: supports constructionist learning — design, test, debug, improve.
- 4 Supports collaboration and inclusion: positive effects on teamwork, confidence, and participation.

Why Python at This Age?

Python minimizes syntax friction and maximizes time on core ideas — other languages become easier once these foundations are solid.

- 1 Readable and beginner-friendly.
- 2 Lets students focus on core programming ideas.
- 3 Real-world and widely used beyond school.
- 4 Works well with MicroPython and device-based projects.

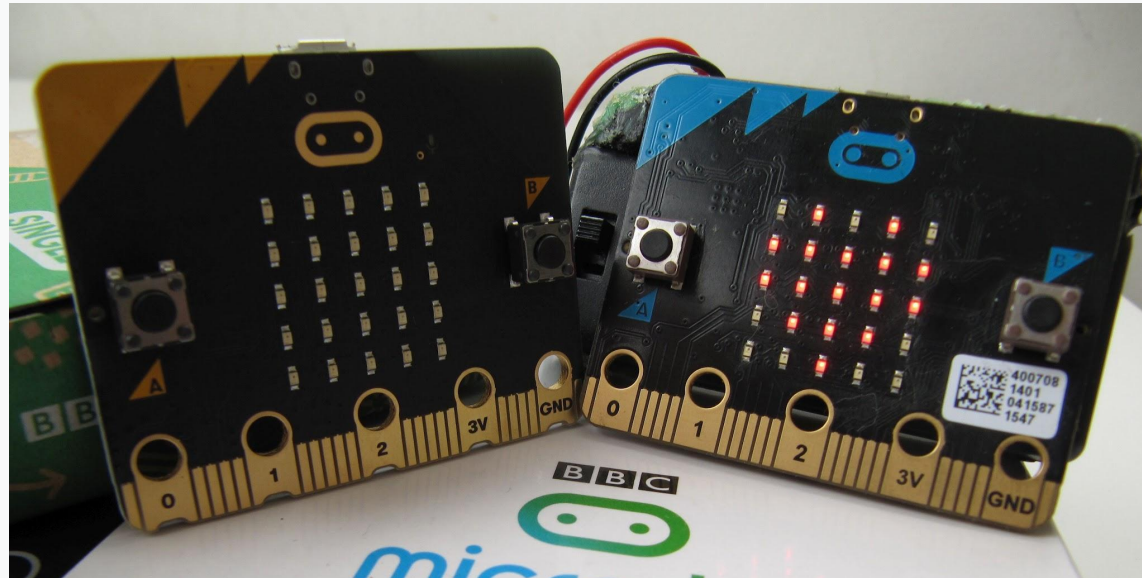
A Gateway, Not a Deep Dive

This workshop is not meant to be a Python deep dive. My goal is to demonstrate how physical computing can be a gateway to learning Python foundational skills.

My Favorite Tool for Physical Computing

Quick show of hands: how many people in the room are familiar with the micro:bit V2?

- 1 Low barrier to entry: affordable, accessible, classroom-ready.
- 2 Immediate feedback: students quickly see code affect lights, buttons, sound, sensors.
- 3 Creative and inclusive: supports ownership, teamwork, broad participation.
- 4 Extensible into robotics: students build first robots, grow into advanced control systems.

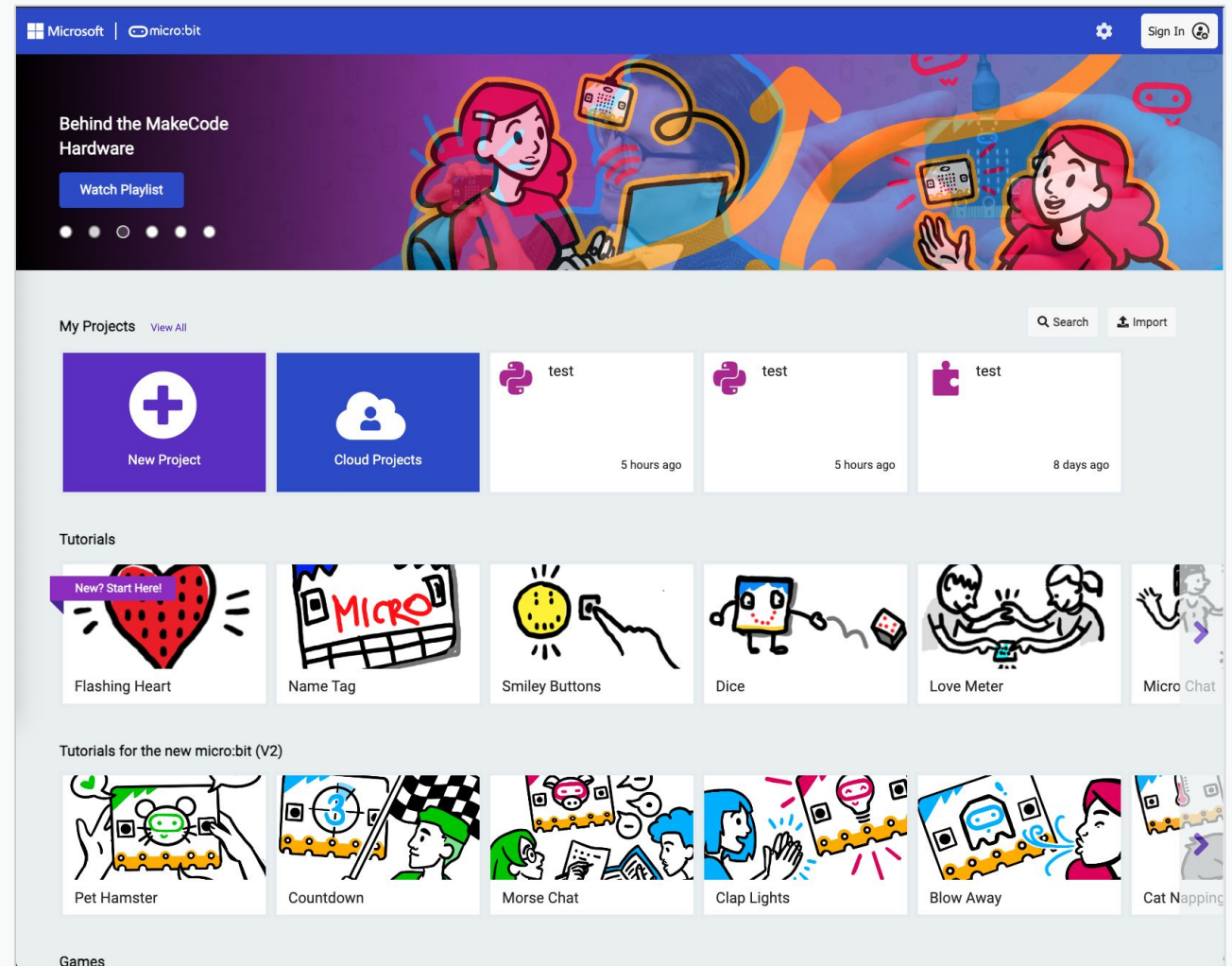


MakeCode & MicroPython

- 1 Free courses and structured pathways for students and teachers.
- 2 Ready-made lessons, projects, and design challenges for classroom use.
- 3 Educator materials: guides, assessments, videos, presentations, workbooks.
- 4 Supports progression from beginner to advanced concepts in a scaffolded way.

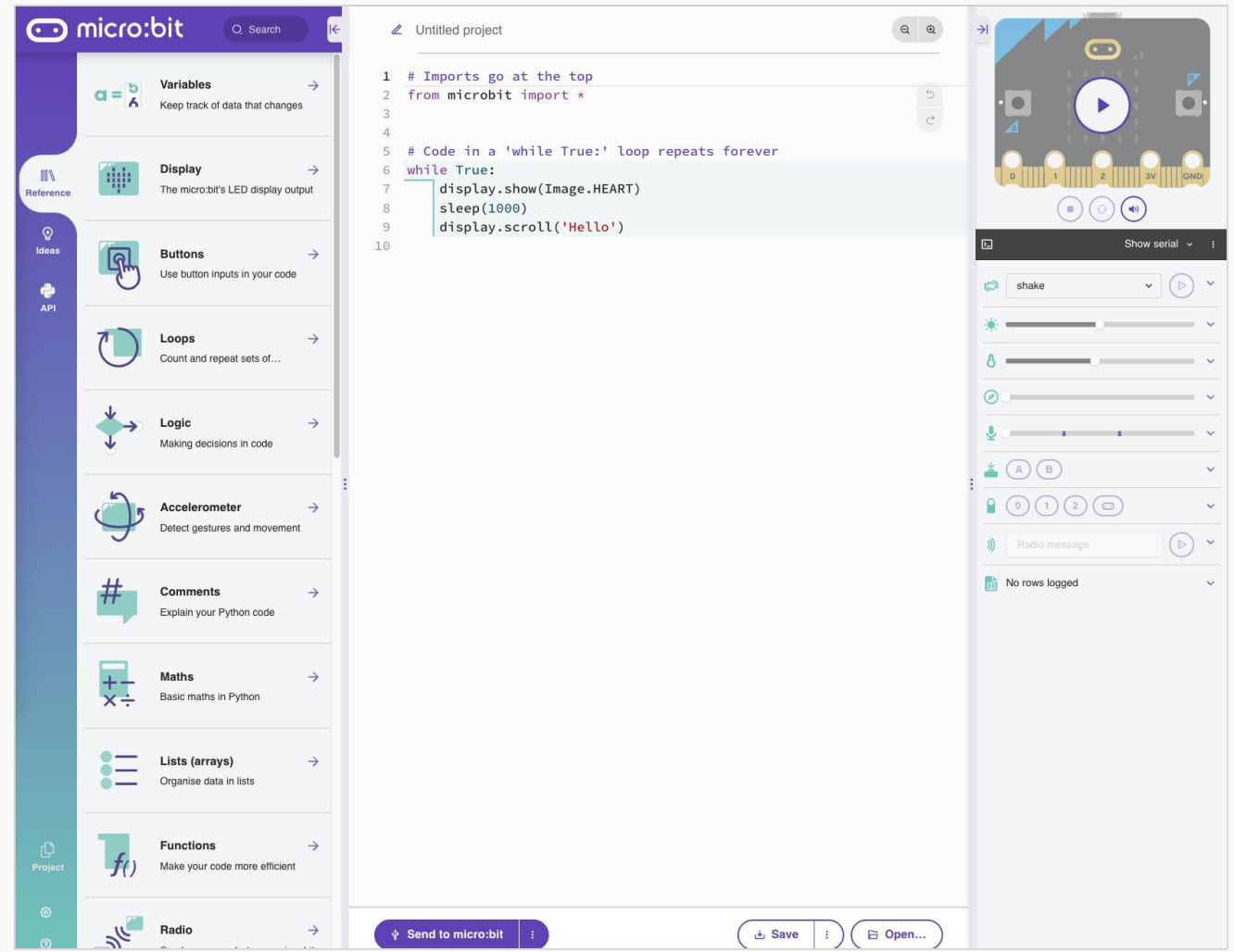
Exercise: Get to Know MakeCode

- 1 Go to makecode.microbit.org.
- 2 Click on the Name Tag tutorial.
- 3 Follow instructions to complete the project.
- 4 Test in simulator, transfer and test on the micro:bit.



Exercise: Get to Know MicroPython

- 1 Go to python.microbit.org.
- 2 Click on Ideas > Emotion Badge.
- 3 Open code file.
- 4 Test in simulator, transfer and test on the micro:bit.



Why Personalization Matters

Students enter with different interests, confidence levels, and ways of making meaning — personalization helps more learners see themselves in the work.

- 1 Some students connect through games, others through music, storytelling, design, or robotics.
- 2 Personalization increases relevance, ownership, and persistence in open-ended making tasks.
- 3 Helps teachers differentiate without abandoning a shared conceptual pathway.
- 4 The goal: not different standards, but different entry points into the same core ideas.

A workspace for today's workshop

I built :bitForge to support today's workshop: you can find, adapt, test and save (in browser only, no accounts) micro:bit challenges.

Find → Test → Adapt → Generate → Review → Prepare



bitforge.indyri.se

The screenshot shows the bitForge website in a web browser. The browser's address bar shows 'bitforge.indyri.se'. The website has a dark header with the 'bitForge' logo and navigation links for 'Home', 'About', and 'Glossary'. Below the header, there's a section for '67 Challenges' with a search bar and filters for 'Concept', 'Component', 'Difficulty Level', and 'CSTA Standard'. A table of challenges is displayed, with columns for 'CHALLENGE', 'CONCEPT', 'LEVEL', 'STATUS', and 'UPDATED'. The challenges listed include 'About Me in Numbers', 'Answer Buzzer', 'Answer Picker', 'Bubble Level', 'Button Tally', 'Callsign Lookup', 'Class Vote', 'Click Counter', 'Countdown Blastoff', 'Cruise Control', 'Doorbell Chime', and 'Emoji Translator'. Each challenge entry includes a brief description, a concept number, a level (e.g., 'Advanced', 'Foundational'), a status (e.g., 'Not tested'), and a date (e.g., 'Jul 18, 2026').

Workshop Outcome

By the end, each of you leaves with:

- 1 One library challenge you have run or examined on a micro:bit.
- 2 One adapted challenge created with a bitForge tool.
- 3 A saved or printed classroom-ready draft.
- 4 A short implementation plan.
- 5 A clearer sense of how this process could fit your teaching.

Today's Agenda

120 minutes / Short demos, partner practice, and open build time

First Half

Welcome and shared foundations

Get oriented with the editors and bitForge

Choose a challenge

Run and test challenge on a micro:bit

Debrief

Break

Second Half

Adapt a challenge with Mad Libs

Explore AI-assisted generation

Review and strengthen a draft

Open build-and-play

Save, reflect, and plan next steps

How We'll Work

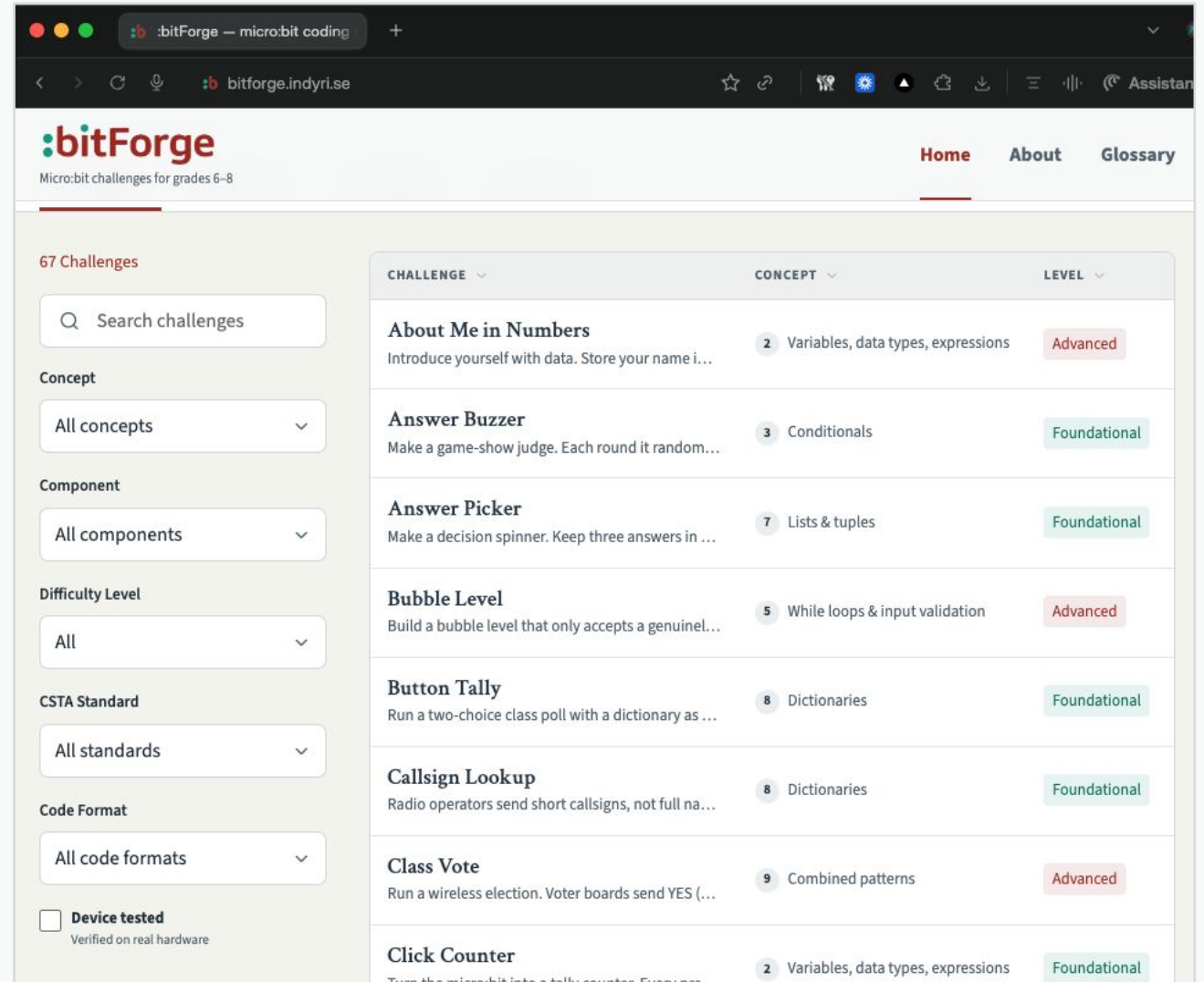
Work with a partner whenever you're on a micro:bit — switch roles halfway.

- 1 Driver: controls the computer and editor.
- 2 Navigator: reads the challenge, checks the result.
- 3 Opening prompt (sticky note): what's one coding concept — or classroom topic — you'd like students to explore with a micro:bit? We'll return to these at closing.

Demo: Challenge Library

MakeCode Python > makecode.microbit.org

MicroPython > python.microbit.org



The screenshot shows the bitForge website, which is a resource for Micro:bit challenges. The page has a dark header with the bitForge logo and navigation links for Home, About, and Glossary. Below the header, there's a section titled "67 Challenges" with a search bar and several filter dropdowns: Concept (All concepts), Component (All components), Difficulty Level (All), CSTA Standard (All standards), and Code Format (All code formats). There's also a checkbox for "Device tested" with the note "Verified on real hardware". The main content area displays a list of challenges in a table format. Each row includes the challenge name, a brief description, a concept number and name, and a difficulty level badge.

CHALLENGE	CONCEPT	LEVEL
About Me in Numbers Introduce yourself with data. Store your name i...	2 Variables, data types, expressions	Advanced
Answer Buzzer Make a game-show judge. Each round it random...	3 Conditionals	Foundational
Answer Picker Make a decision spinner. Keep three answers in ...	7 Lists & tuples	Foundational
Bubble Level Build a bubble level that only accepts a genuinel...	5 While loops & input validation	Advanced
Button Tally Run a two-choice class poll with a dictionary as ...	8 Dictionaries	Foundational
Callsign Lookup Radio operators send short callsigns, not full na...	8 Dictionaries	Foundational
Class Vote Run a wireless election. Voter boards send YES (...)	9 Combined patterns	Advanced
Click Counter Turn the micro:bit into a tally counter. Every time...	2 Variables, data types, expressions	Foundational

Exercise: Pick & Practice a Challenge

With your partner, find one challenge that:

- 1 Practices a concept relevant to your students.
- 2 Uses hardware available at your table.
- 3 Has an appropriate difficulty level.
- 4 Can fit into a lesson you teach.

Micro:bit Test Drive: Two Routes

Starter Route

Open the provided solution, transfer it to the micro:bit, change one visible behavior (image, message, threshold, sound), and test the change.

Builder Route

Begin from the student prompt, implement one success criterion, test it on the device, and use the provided solution only if needed.

Choose the route that best matches your comfort level - you can switch any time.

Debrief: Update the Challenge Tracker

That experience is part of teacher review — code that looks reasonable on screen can behave differently once sensors, timing, or sound are involved.

- 1 Worked as expected.
- 2 Needed debugging.
- 3 Could not complete because of setup.
- 4 Would need more scaffolding for my students.

10-Minute Break

We'll pick back up with the Mad Libs demo

Demo: Mad Libs

Adapt a challenge using existing patterns

- 1 Concept: Conditionals
- 2 Context: Group-work volume
- 3 Input: Microphone
- 4 Output: Quiet/loud icon
- 5 Level: Foundational

The screenshot shows the bitForge website interface for creating a challenge. The browser address bar shows 'bitforge.indyri.se / :bitForge'. The website has a navigation bar with 'Home', 'About', and 'Glossary'. Below this is a secondary navigation bar with 'Library', 'Mad Libs' (highlighted), 'Generate', and 'Resources'. The main content area is divided into two columns.

CHALLENGE SETUP

Make it your own Surprise me

Type or choose a suggestion ☐ Fixed choice

I want students to practice Conditionals .

They will build noise-level monitor as

a group-work volume indicator .

The project will use the current microphone reading to

display a quiet or loud icon .

Make it foundational .

Variation ID BF-BRIGHT-274 Type your own wording or choose a suggestion - browser-only

CHALLENGE PREVIEW Variation ID BF-BRIGHT-274

Noise-level monitor

CONCEPT Conditionals **DIFFICULTY** Foundational

HARDWARE
1 micro:bit (v2) · sensors, led

Challenge
Build a group-work volume indicator. Read the current microphone reading repeatedly and use a conditional to display a quiet or loud icon. Test every range using real classroom sounds.

Success criteria

- The program samples sound repeatedly.
- A conditional selects the status.
- Every defined range can be demonstrated during testing.

Optional extension
Add a third status range.

Review and test this challenge before classroom use.

Print / save PDF Download Markdown Download data

Exercise: Create a Quick Variation

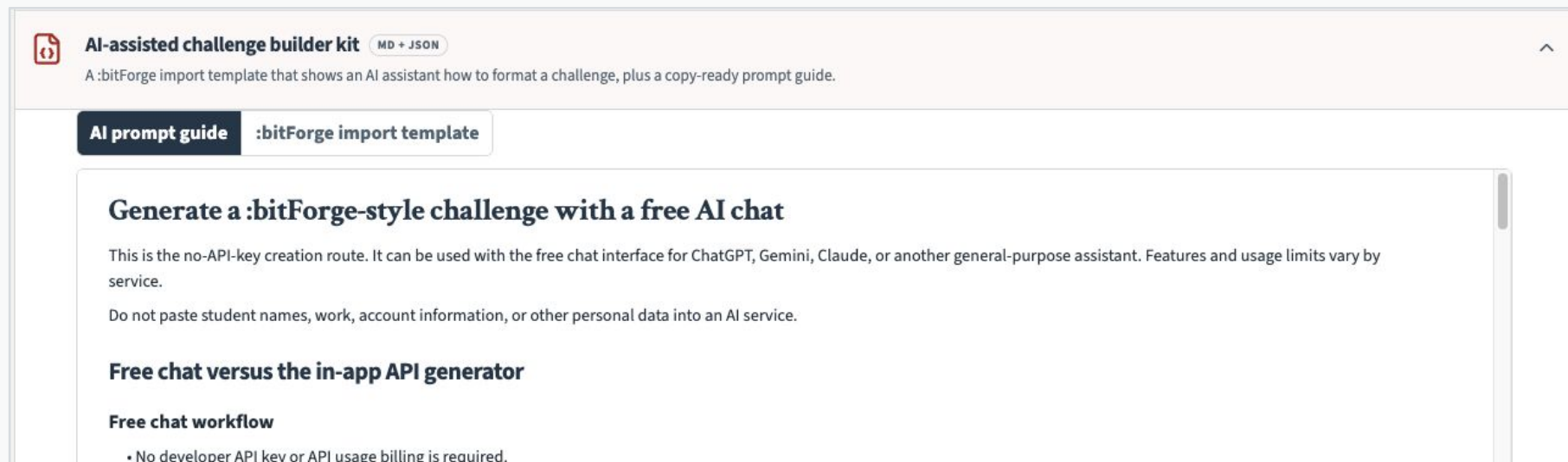
Create a Mad Libs variation of the challenge you tested, changing at least one context, one input/output, and the difficulty if appropriate. Compare the original challenge and the Mad Libs variation with your partner. Discuss:

- 1 Mad Libs helped me by...
- 2 The variation fits my learners better because...
- 3 Before using the generated variation, I would change...
- 4 The goal is not to redesign everything, but to make any challenge more relevant to a different group of learners.

Demo: AI-Assisted Generation

Useful when concept, hardware, time, and context fall outside existing patterns. Specificity matters: time, grouping, prior knowledge, editor, and hardware are instructional constraints, not decoration.

- 1 Open Resources > AI-assisted challenge builder kit
- 2 Use the guided prompt in ChatGPT, Claude, Gemini etc. Download and attach the .json template.
- 3 Have these details ready: coding concept and difficulty, grade level and prior knowledge if applicable, lesson time and grouping, editor and available hardware
- 4 Treat the AI output as a draft. Review the instructions and code, then test it on the micro:bit.



Privacy & Review

The in-app route uses your OpenAI or Anthropic API key, held only in the browser tab and is not stored.

- 1 Use a restricted project key.
- 2 Never include student names, student work, or personal information.
- 3 Review sequence: instructional fit, reading level, observable success criteria, code inspection, hardware testing.
- 4 No-API alternative: the Resources area includes a prompt guide for a free chat interface; its JSON result imports into bitForge for validation.

Exercise: Review AI Output

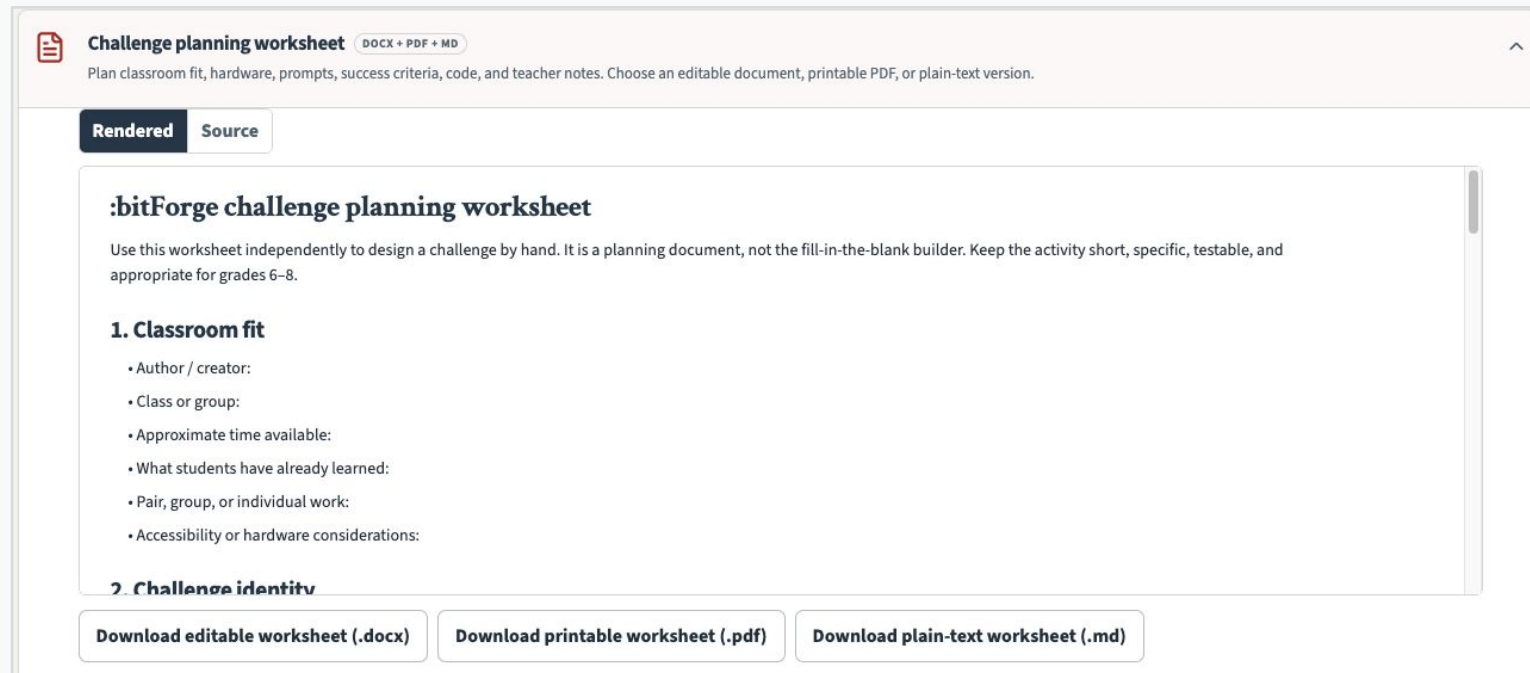
Use this framework to evaluate the AI Output

- 1 Instructional fit: does it match the concept, learners, time, and hardware?
- 2 Reading level: are the generated directions age appropriate?
- 3 Observable success criteria: Can you and your students clearly tell whether it worked?
- 4 Code inspection: is the code correct, understandable, and aligned to the intended concept?
- 5 Hardware testing: does it actually work on the micro:bit?

Demo: Challenge Planning Worksheet

Use this worksheet when you need more control, or when you want to shape the challenge before generating or coding.

- 1 Mad Libs > *Quick variations*.
- 2 AI-assisted generation > *Personalizing existing patterns*.
- 3 Challenge planning worksheet is the most deliberate route when you want to design from your classroom context first.



The screenshot shows a web interface for a "Challenge planning worksheet". At the top, there's a header with a document icon, the title "Challenge planning worksheet", and a small button "DOCX + PDF + MD". Below this is a subtitle: "Plan classroom fit, hardware, prompts, success criteria, code, and teacher notes. Choose an editable document, printable PDF, or plain-text version." The main content area has two tabs: "Rendered" (selected) and "Source". Under the "Rendered" tab, the title ":bitForge challenge planning worksheet" is displayed. Below the title is a paragraph: "Use this worksheet independently to design a challenge by hand. It is a planning document, not the fill-in-the-blank builder. Keep the activity short, specific, testable, and appropriate for grades 6–8." The content is organized into sections. The first section is "1. Classroom fit", which includes a list of bullet points: "• Author / creator:", "• Class or group:", "• Approximate time available:", "• What students have already learned:", "• Pair, group, or individual work:", and "• Accessibility or hardware considerations:". The second section is "2. Challenge identity". At the bottom of the interface, there are three buttons: "Download editable worksheet (.docx)", "Download printable worksheet (.pdf)", and "Download plain-text worksheet (.md)".

Challenge planning worksheet DOCX + PDF + MD

Plan classroom fit, hardware, prompts, success criteria, code, and teacher notes. Choose an editable document, printable PDF, or plain-text version.

Rendered Source

:bitForge challenge planning worksheet

Use this worksheet independently to design a challenge by hand. It is a planning document, not the fill-in-the-blank builder. Keep the activity short, specific, testable, and appropriate for grades 6–8.

1. Classroom fit

- Author / creator:
- Class or group:
- Approximate time available:
- What students have already learned:
- Pair, group, or individual work:
- Accessibility or hardware considerations:

2. Challenge identity

Download editable worksheet (.docx) Download printable worksheet (.pdf) Download plain-text worksheet (.md)

Exercise: Use the Challenge Planning Worksheet

Choose either your Mad Libs variation or the AI-assisted draft. Use the Challenge Planning Worksheet to strengthen one area that still feels under developed. Exchange sheets with your partner and ask:

- 1 What would students be doing first?
- 2 How will success be observed?
- 3 Where might students need a little support?
- 4 What should be tested on the micro:bit before class?
- 5 What is one small change that could make the activity fit your learners better?

Choose Your Direction

Choose one challenge or draft to test or improve during open build time

- 1 The original Library challenge
- 2 Your Mad Libs variation
- 3 Your AI-assisted draft
- 4 Your Challenge Planning worksheet

Open Build-and-Play Period

Four Ways to Continue

Work in pairs with micro:bits. Choose the pathway that best matches what you want to work on next.

- 1 Make it work: run a selected challenge and satisfy one success criterion.
- 2 Make it yours: change the story, output, images, sounds, or controls — keep the coding concept.
- 3 Make it teachable: improve the prompt, add scaffolding, create starter code.
- 4 Make it stretch: add an extension for early finishers.

Things To Consider As You Work

- 1 What did the physical device reveal?
- 2 Is your adaptation still practicing the intended concept?
- 3 Which success criterion are you testing?
- 4 What would you demonstrate before releasing students?
- 5 Where could students make a productive mistake?
- 6 What support preserves the challenge without solving it for them?

Reflection

How was your experience, what did you learn, how would you change it?

The Process

What felt immediately useful?

Where did you hesitate or get confused?

Which approach best supported your thinking?

What would make this process easier to use?

Workshop Experience

Which activity best helped you understand how physical computing could support your students?

Where did you need more or less time?

Was the micro:bit setup manageable?

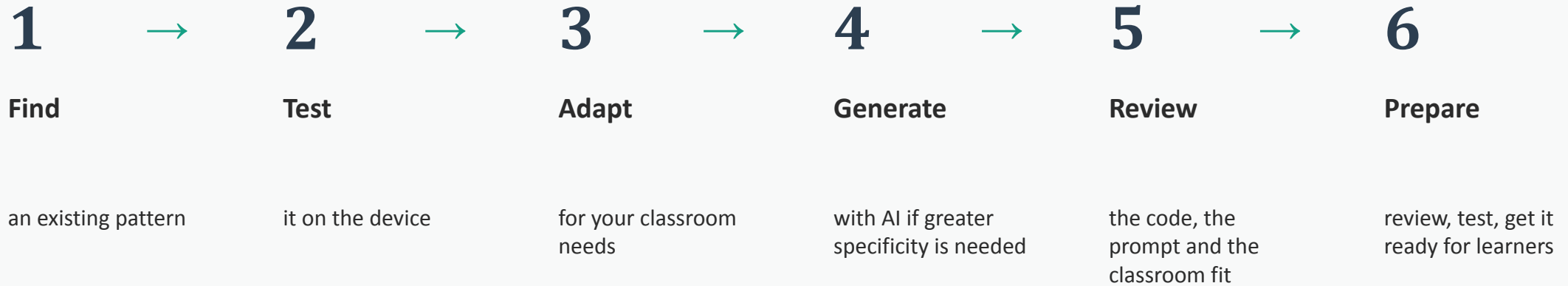
What should change for the next workshop?

Select or create one activity that fits your learners — and can be tested with confidence.

Save your work, submit your exit ticket, and take any notes or drafts you want to revisit



The Progression, However You Start



Thank you!

Email: rucha@indyri.se

LinkedIn: <https://www.linkedin.com/in/ruchagokhale>

Portfolio: <https://rucha.indyri.se>