Assignment 1: Hello, Satellite!

Mission briefing: open a radio channel and prove your satellite is listening.

Your mission

You've just been hired as mission control for a spy satellite. Before any spy work begins, you need one thing: a working radio.

Build your first block program. Make the satellite print a message to the mission log, something like calling mission control. Then run a simulation and read the log.

You succeed when:

1. Your message shows up in the log.
2. You can say **how many times** it showed up, and **why** it's that number.

Here's the secret you are about to discover. Your program has two pages. The **init** page runs once, when the match starts. The **main** page is different:

> ⏱ **Every second:** the satellite reads your main page again, top to bottom, and runs every block on it. Then it does the whole page again the next second. Your blocks are not a one-time story. They are a checklist the satellite checks every second.

Mission rules

- Create the project with game **Free Mode** and the **Graphical Editor**.
- Use blocks from the **Debug** category only.
- Simulate with the **default settings**.
- Your message must appear in the log, and you must be able to say HOW MANY times it appeared and why.
- Bonus experiment: move your block to the **init** page and simulate again. Predict the log before you run it.

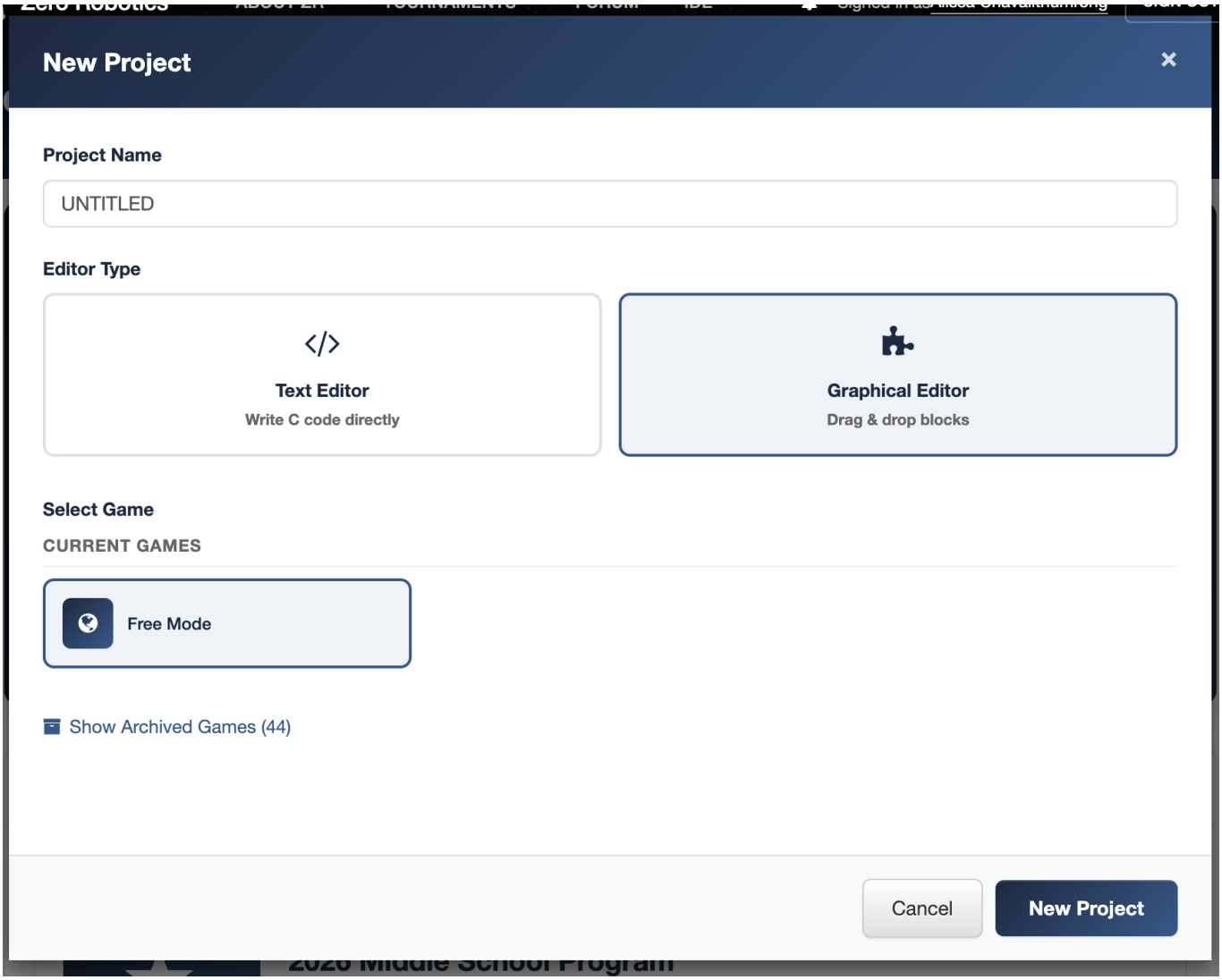
Blocks to explore

- **Debug** is the only category you need today. One block in here prints a message to the log, and a small text block holds the words. Type your message between the two quote marks.

Build it, step by step

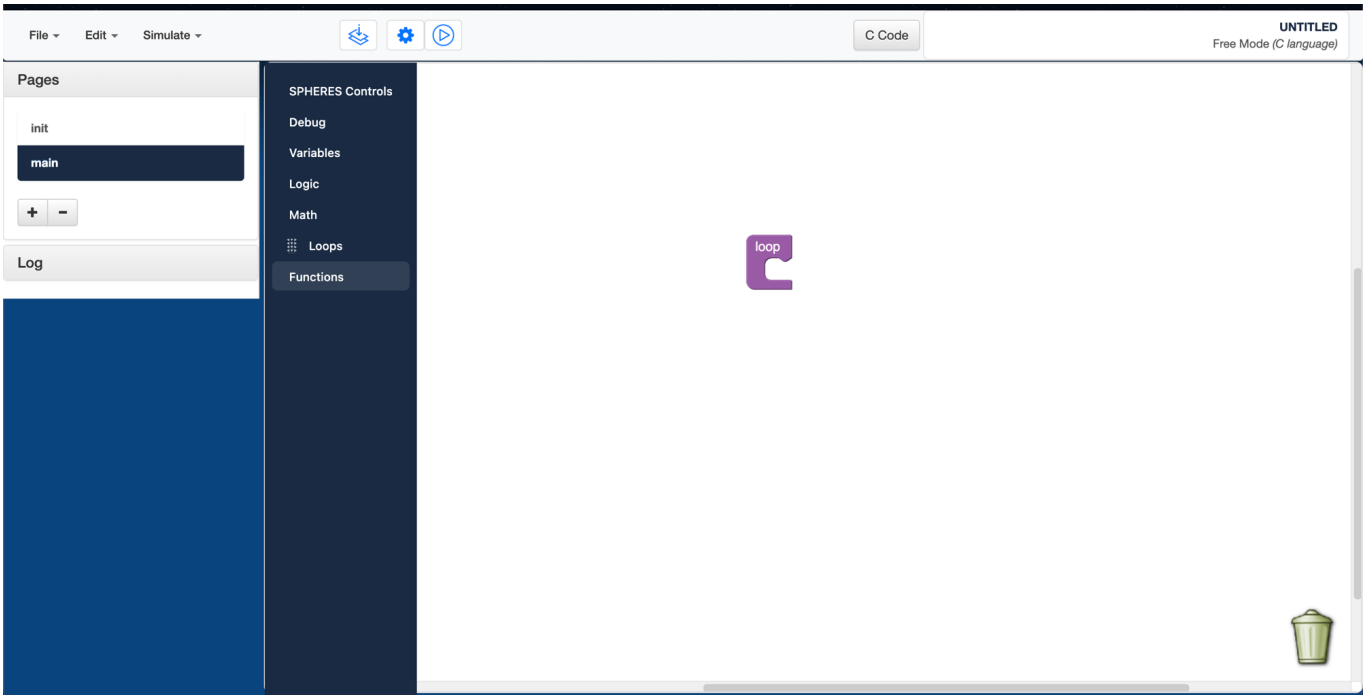
Follow these steps in order. Each picture shows exactly what you should see on screen.

1. Create the project



Make a new project. In the dialog, pick the game **Free Mode** and the **Graphical Editor**, give your project a name, and create it.

2. Meet your workspace



The block palette is on the left, sorted into categories. Notice the page tabs: your program has an **init** page and a **main** page. Click between them once, just to see both. Then make sure you are on the **main** page.

3. Find the main page's root block

The main page comes with this **loop** root block already in place. You can't delete it or move it; it is the page itself. Everything you snap inside its slot runs **once per second, every second**, for the whole match.

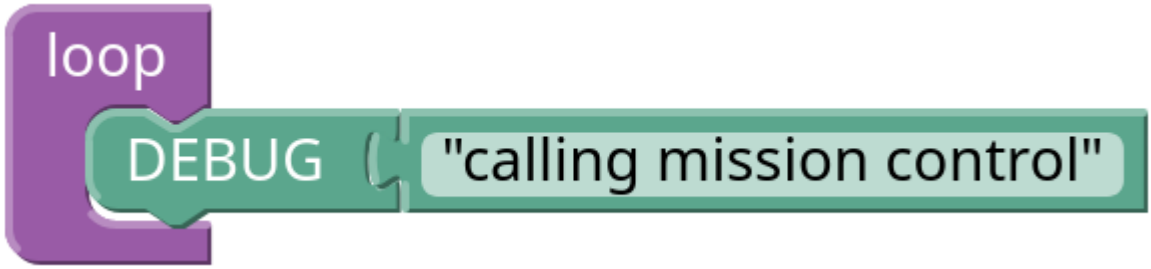
4. Drag out the DEBUG block

Open the **Debug** category in the palette and drag the `DEBUG` block into the loop root's slot. Snap it inside. This block prints whatever is plugged into its socket to the mission log. Right now the socket is empty. It has nothing to say yet.

5. Plug in the text block and type your message

From the same **Debug** category, drag out the small text block and plug it into the `DEBUG` block's socket. Click between the two quote marks and type your message: **calling mission control** (or your own words; any message works).

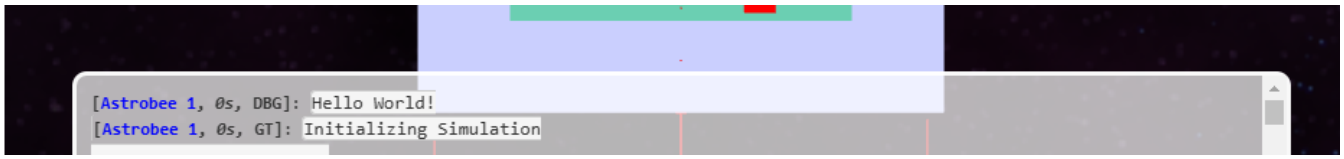
6. Compare your workspace



Your main page should now look exactly like this: the loop root block holding one `DEBUG` block, with your message inside the quotes. The init page stays empty. That's the whole program: one block.

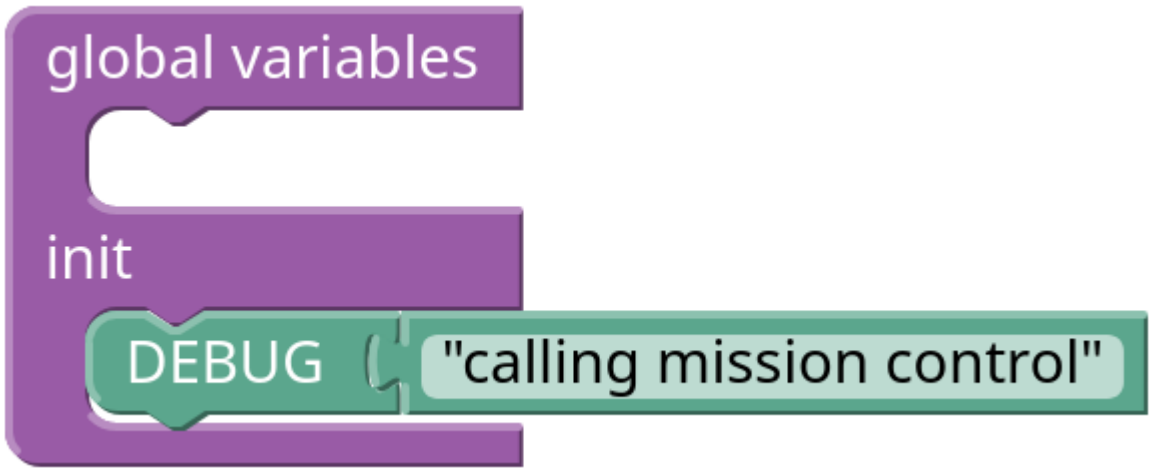
7. Simulate and read the log

Run a simulation with the **default settings**. While it runs (or after), open the log:



Find your message. Now count: **how many times does it appear?** In a 60-second simulation it should be about 60 lines, one for every simulated second. Before you move on, say why it's that number out loud.

8. The experiment: move your block to the init page



Drag your `DEBUG` block (with its text still attached) off the main page and snap it into the **init** page's "init" slot, like the picture. The main page's loop slot is now empty. **Before you simulate: predict what the log will show.** Then run it and check. Your message should appear **exactly once**, right at the start, because the init page runs once, when the match starts.

Hint

The main page is not read once. The satellite reads it again every second of the match. Your block does not need to repeat itself; the page it sits on already does.

Check your work

- With your block on the **main** page: simulate, open the log, and you should see your message repeated, **one line for every simulated second** (about 60 lines in a 60-second simulation). Count them!
- With the same block moved to the **init** page's "init" slot: simulate again, and your message appears **exactly once**, right at the start.
- Try the **"C Code"** toggle. Your message is sitting inside something called `void loop()` when the block is on the main page, and inside `void init()` when it's on the init page. Same program, two costumes.

Reflection (write 3–5 sentences)

Explain your block program in your own words, as if to a teammate who missed class: **what does your program do every second?** Then answer this: how many times did your message appear in the log, and why exactly that number?