PRESENTATION 7 · LEARNING WITH STATEMENTS

One of each

Iron, silver and gold, and a robot that counts what it collects using variables.

TheBotCup 2026 · AiBotCup Simulator



RECAP · PRESENTATION 6

Four statements found all the iron

IronLeft
dark on the left → Collect

IronRight
dark on the right → Collect

Ahead
wall close → turn left

Forward
otherwise drive

That robot collected **everything** it found. Today that is not good enough: it must **choose**.

PART 1

The mission: Session 7

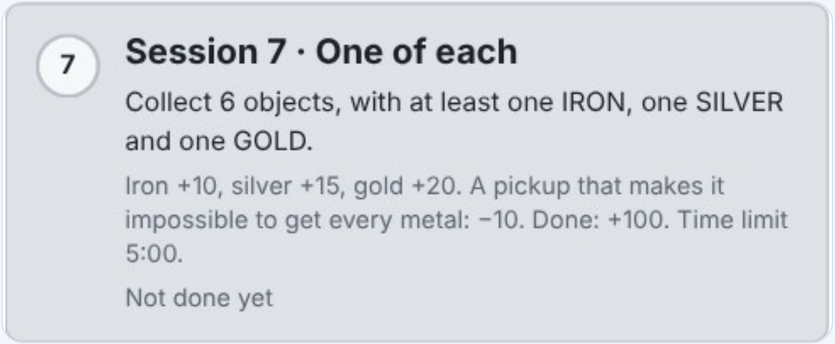
Six objects, every metal at least once

SESSION 7 · ONE OF EACH

Collect 6, with at least one of each metal

What happens	Points
Iron · silver · gold	+10 · +15 · +20
All 6 done, every metal	+100
A pickup that makes it impossible	−10, refused

Time limit **5:00**. The robot holds at most **6** objects.





THE FIELD

Three metals, mixed up

- **7 iron** · black
- **6 silver** · grey
- **6 gold** · yellow

Same field as Session 6: block in the middle, START at the bottom.

COLOUR SENSORS · FROM PRESENTATION 6

Tell the three metals apart

Metal	R	G	B
Iron	0 – 60	0 – 60	0 – 60
Silver	90 – 135	92 – 137	98 – 142
Gold	195 – 255	135 – 200	0 – 80
Floor	about 226	about 232	about 238

These ranges come from the simulator’s own tip. Floor never fits any of them.

PART 2

The main problem: counting

How many do I have already?

THE PROBLEM

Six places, and they can fill up wrong



Gold is still missing and only **one place** is left. If the next object is not gold, the mission becomes impossible: the simulator **refuses it, −10**.

The robot must know **how many of each** it already has.

WE TESTED IT · NO COUNTING

“Collect everything” lost points

What it did

Picked up 3 iron and 2 silver first. Then **one pickup was refused** because gold was still missing.

Score

It finished, but with **170** instead of 180: a −10 for the refused pickup.

On another field, or with other luck, it could be worse. The robot needs a **memory**.

THE IDEA

To count, the robot needs a number it can remember and change.

Sensors tell what is happening **now**. A counter remembers what happened **before**. In programming that is called a **variable**.

PART 3

Variables

The robot’s memory

WHAT IS A VARIABLE?

A labelled box that holds one number

It has a **name** (Iron) and a **value** (0, 1, 2...).

The program can **read** the value and **change** it.

It **keeps** its value every 60 ms tick, until you change it.



HOW IT WORKS · THE COUNTER

Read it in a condition, change it in an action

READ · Advanced Condition

```
Iron < 2
```

True while Iron is 0 or 1.

CHANGE · Advanced Action

```
Iron = Iron + 1;
```

“New Iron is old Iron plus one.”

Iron: 0 → 1 → 2 … then **Iron < 2** is false: no more iron.

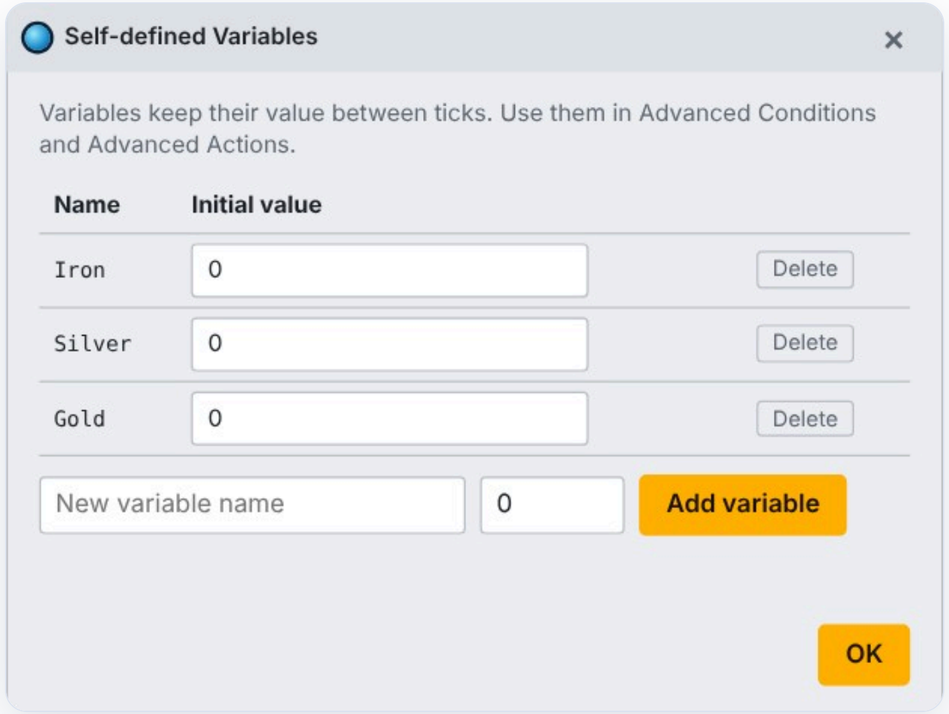
MAKE VARIABLES IN STATEMENTS

The Var button



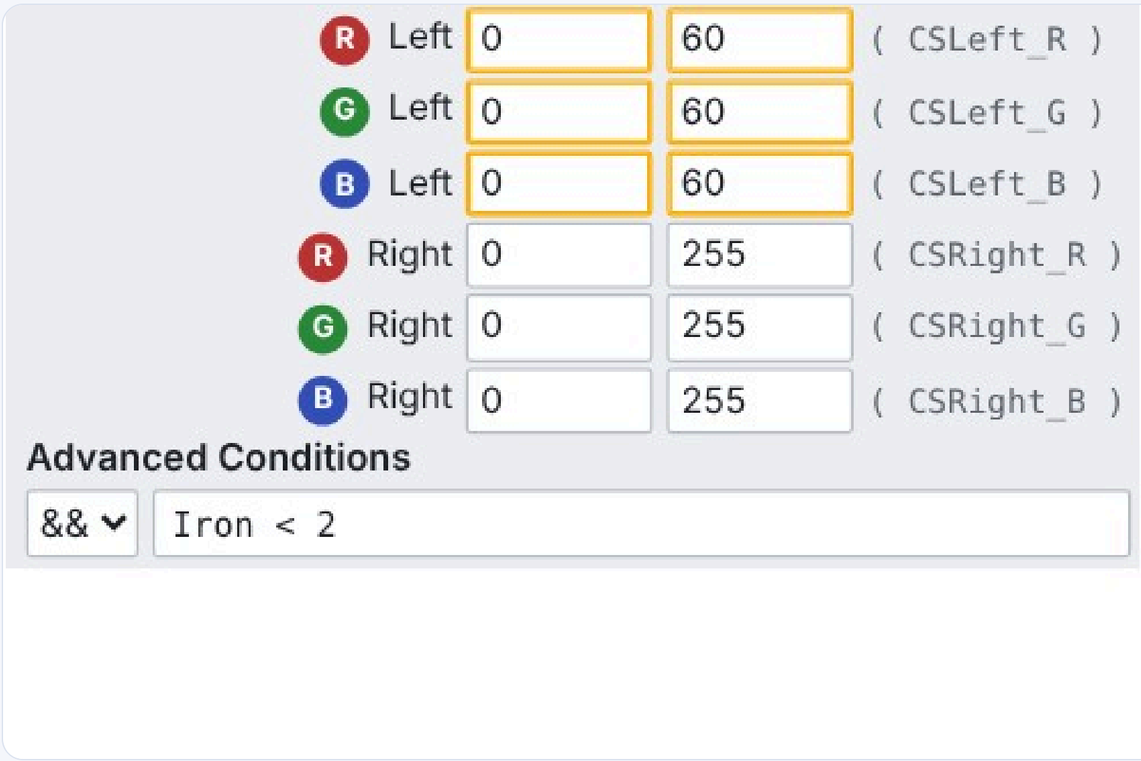
1. In the AI panel press **Var**.
2. Type a name, keep initial value **0**.
3. Press **Add variable**. Repeat, then **OK**.

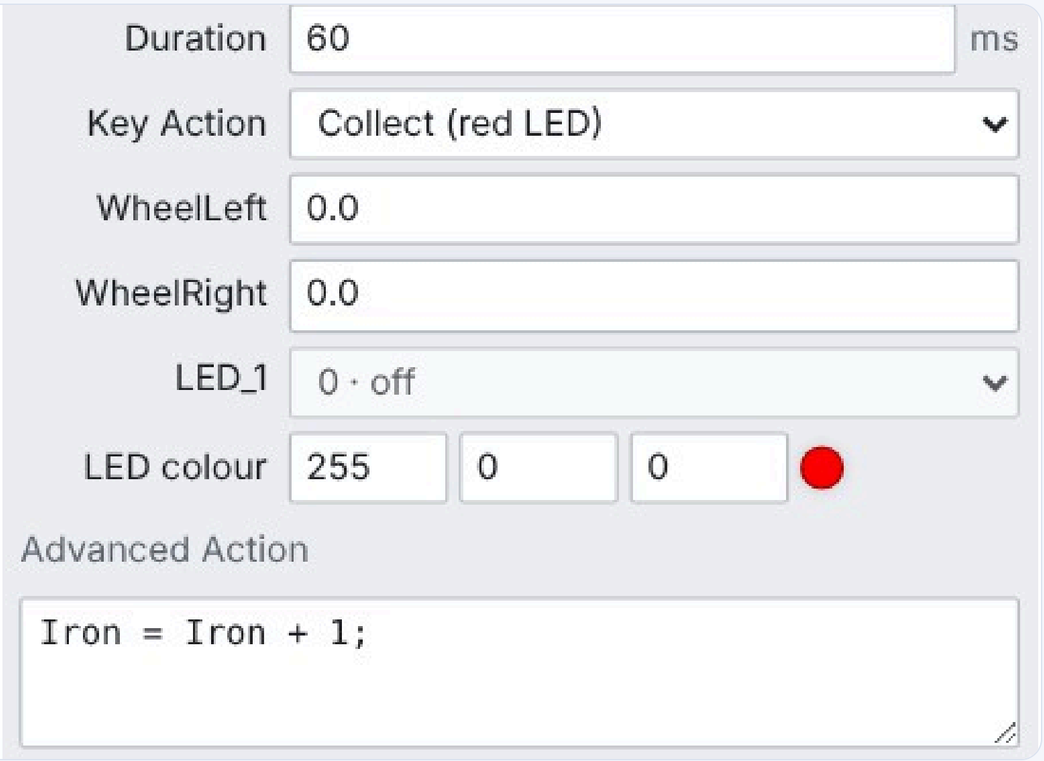
Names: letters, digits and _, starting with a letter. A sensor name cannot be used.



USE VARIABLES IN A STATEMENT

Two new boxes at the bottom of the panel





Advanced Conditions (left): **&&** means “and”: the ranges **and** Iron < 2 must be true.

Advanced Action (right): runs **once** each time this statement starts.

OUR RULE

Collect a metal only while its counter is below 2

2 + 2 + 2 = 6

Six objects can only happen as two of each, so there is **always one of each** and no pickup is ever refused.

PART 4

Step by step in statements

Three counters, eight statements

STEP 1 · VARIABLES, THEN THE WHOLE PROGRAM

Make Iron, Silver, Gold = 0, then 8 statements

Statement	Ranges	Adv. Condition	Action	Adv. Action
1 IronLeft	CSLeft iron	Iron < 2	Collect	Iron = Iron + 1;
2 IronRight	CSRight iron	Iron < 2	Collect	Iron = Iron + 1;
3 SilverLeft	CSLeft silver	Silver < 2	Collect	Silver = Silver + 1;
4 SilverRight	CSRight silver	Silver < 2	Collect	Silver = Silver + 1;
5 GoldLeft	CSLeft gold	Gold < 2	Collect	Gold = Gold + 1;
6 GoldRight	CSRight gold	Gold < 2	Collect	Gold = Gold + 1;
7 Ahead	US_Front 0–20	—	−2 / 2, 700 ms	—
8 Forward	none	—	4.5 / 4.5	—

STEP 2 · STATEMENTS 1 AND 2 · IRON

IronLeft, then copy it for the right side



IronRight: the same, but the 0–60 ranges go on **CSRight** R, G and B. Same counter: Iron.

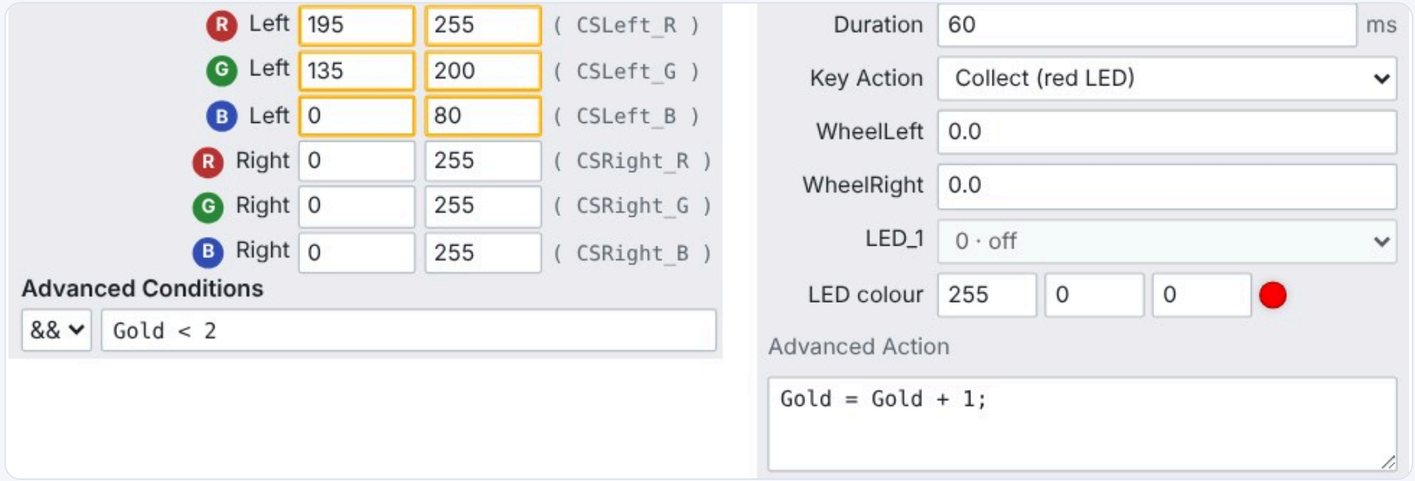
STEP 3 · STATEMENTS 3 TO 6 · SILVER AND GOLD

Same pattern, new ranges, own counter

3 SilverLeft
Silver < 2



5 GoldLeft
Gold < 2



STEP 4 · STATEMENTS 7 AND 8

Ahead and Forward, as in Presentation 6

7 Ahead
US_Front 0–20

Front	0	20	(US_Front)
Left	0	255	(US_Left)
Right	0	255	(US_Right)

Duration	700	ms
Key Action	None	
WheelLeft	-2.0	
WheelRight	2.0	

8 Forward
no conditions

Front	0	255	(US_Front)
Left	0	255	(US_Left)
Right	0	255	(US_Right)

Duration	60	ms
Key Action	None	
WheelLeft	4.5	
WheelRight	4.5	

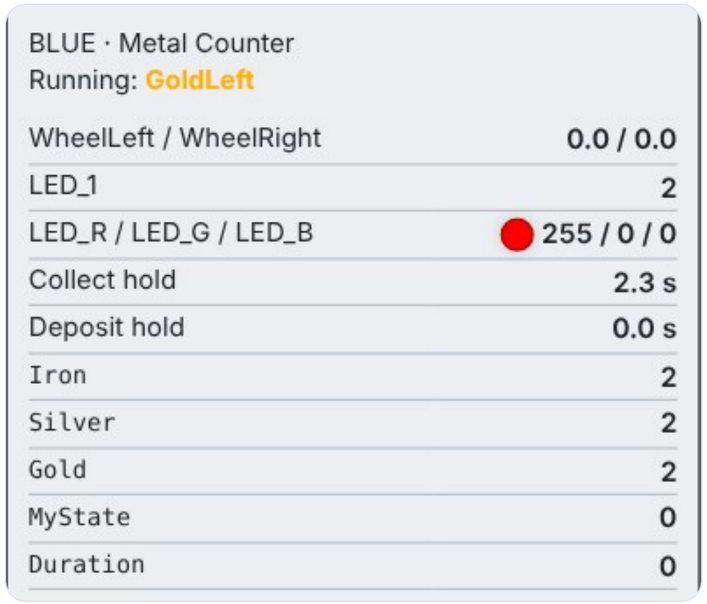
Collect statements stay on top. A metal the robot no longer wants simply matches nothing, so **Forward drives over it**.

STEP 5 · WATCH THE VARIABLES

Press Debug: your counters are listed live



7.8 s · first iron



74 s · last gold

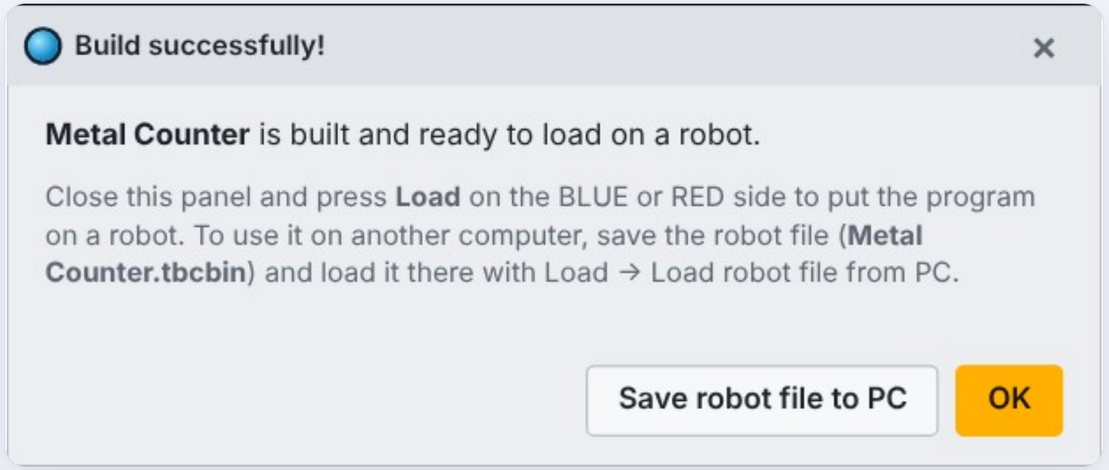
The counter goes up **as soon as Collect starts**.

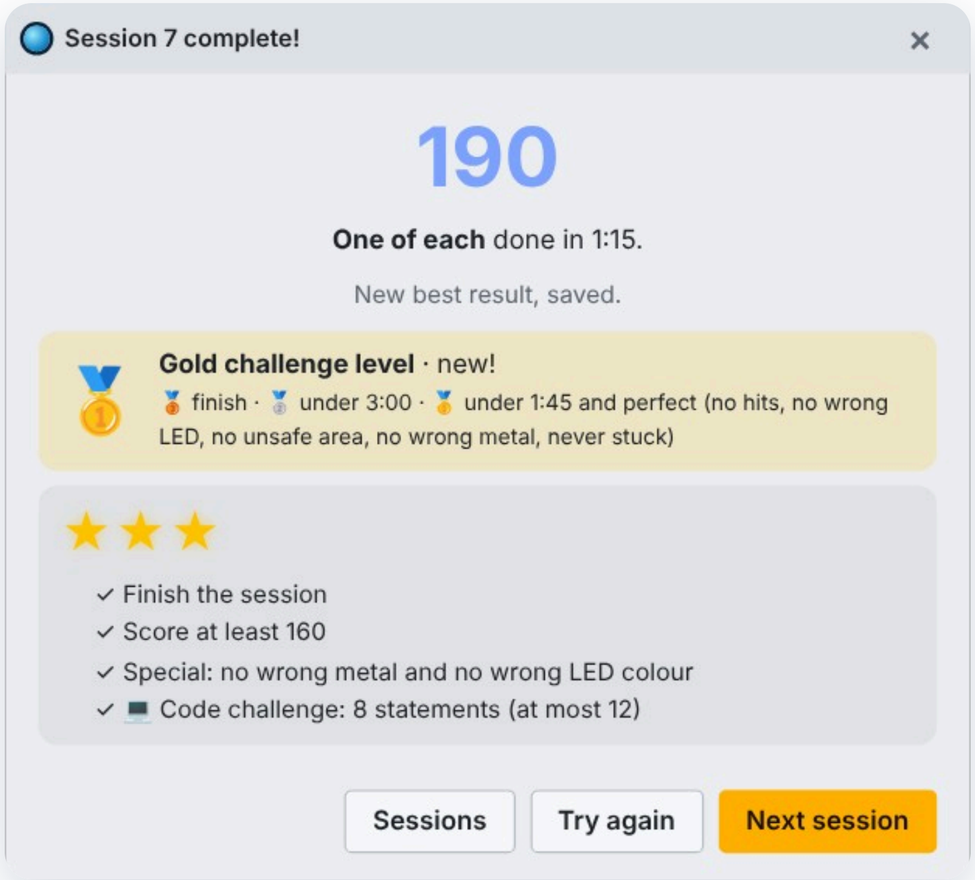
At the end: **2, 2, 2**.

BUILD AND LOAD

Name it, build it, load it

1. Name the program **Metal Counter**.
2. Check: **3 variables**, 8 statements in order.
3. Press **Build**, then **OK**.
4. Press **Load** on the BLUE side.





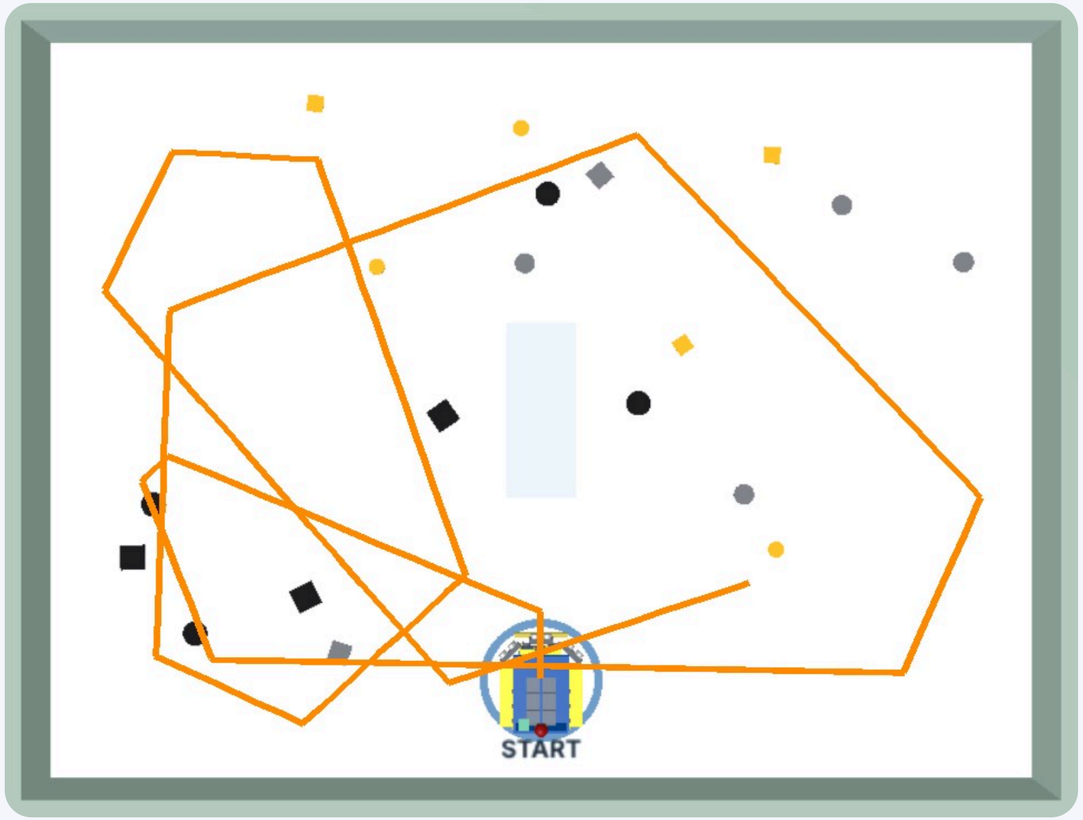
RESULT · TESTED

190 points in 1:15 · Gold

Score · 2×10 + 2×15 + 2×20 = 90, plus 100 = **190**.

Medals · Silver: under 3:00 · Gold: under 1:45 and perfect.

Stars · no wrong metal · code challenge: 8 of at most 12.



THE PATH · RECORDED

Two of each, then done

Order in our run: iron, silver, silver, iron, gold, gold.

Once a counter reaches 2, that metal is ignored.

The counters let the robot **choose**.

DID NOT WORK?

Check first, then tune carefully

What you see	What to do
Build error in a statement	Exact spelling (Iron, not iron) and the **;** at the end
Counter never changes in Debug	Advanced Action must say **Iron = Iron + 1;**
Collects a third of one metal	Advanced Condition missing, or **&&** set to **||**
Never collects silver or gold	Check the R, G, B ranges against the colour table
Old program still on the robot	**Build** again, then **Load** on BLUE

NEXT TIME

Presentation 8 · Two of each

Exactly two of every metal, and the simulator’s **own built-in counters**.