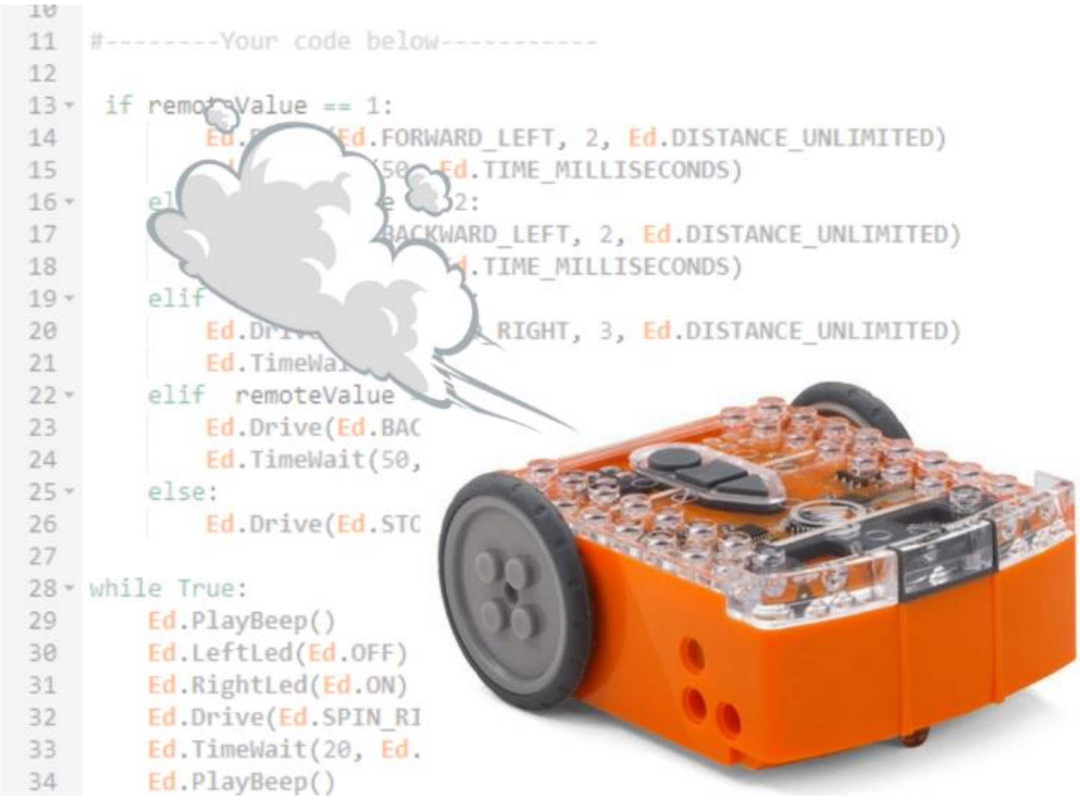
Edison and EdPy: Get Edison moving

Teacher's notes



```
#--------Your code below-----------

 if remo  Value == 1:
        Ed.       Ed.FORWARD_LEFT, 2, Ed.DISTANCE_UNLIMITED)
                 50,  Ed.TIME_MILLISECONDS)
    el           e    2:
                   BACKWARD_LEFT, 2, Ed.DISTANCE_UNLIMITED)
                         .TIME_MILLISECONDS)
    elif
        Ed.Dr               RIGHT, 3, Ed.DISTANCE_UNLIMITED)
        Ed.TimeWa
    elif  remoteValue
        Ed.Drive(Ed.BAC
        Ed.TimeWait(50,
    else:
        Ed.Drive(Ed.STC

while True:
    Ed.PlayBeep()
    Ed.LeftLed(Ed.OFF)
    Ed.RightLed(Ed.ON)
    Ed.Drive(Ed.SPIN_RI
    Ed.TimeWait(20, Ed.
    Ed.PlayBeep()
```

Contents

Go ahead – show off!

We love seeing how classrooms use Edison! If you and your students want to share your Hour of Code Edison EdVenture, be sure to tag us into the fun!

@meetedison	twitter.com/meetedison
@meet_edison	instagram.com/meet_edison
@meetedison	facebook.com/meetedison

About this lesson and guide

This guide offers teachers and instructors overview information, facilitation recommendations and other supporting information for the *Edison and EdPy: Get Edison moving* lesson available at https://meetedison.com/robotics-lesson-plans/get-edison-moving-in-edpy/

Do you need to read this whole guide to run the lesson? **Absolutely not!**

Once the robots and programming devices are set up[1], you can start learning along with your students! The student sheets for this lesson have been designed to allow students to work through the different parts of the lesson independently, learning about the Edison robot, how to use the Edy programming environment, and the key computer science learning objectives of the lesson. This guide simply offers further information for teachers and instructors to help make using this lesson easy and fun.

Each section of the lesson is included in this guide along with any relevant supporting information for that part. Supporting information is divided into the following sections:

Overview
General information about the section and key learning objectives covered.

Delivery recommendations
Suggestions for how you can cover the lesson section if you want to run the lesson in a more facilitator-led capacity.

Tips and tricks
Helpful hints and ways to overcome common issues students may encounter.

Creative Commons licence attribution details

Developed and written by: Kat Kennewell

[1] The *Getting started with Edison and EdPy* set available at https://meetedison.com/robotics-lesson-plans/get-edison-moving-in-edpy/ has step-by-step help for setting up your robots and programming devices. If you are new to Edison or EdPy, it is recommended you start with that guide.
[2] https://meetedison.com/robot-programming-software/edpy/#edpy-resources
[3] https://meetedison.com/robot-programming-software/edscratch/#EdScratch-resources
[4] http://creativecommons.org/licenses/by-sa/4.0/

Lesson overview

Introduce text-based programming and the key computational concept of sequence to your students using Edison robots and EdPy[5], a text-based programming language based on Python. Students learn computer science and programming basics before taking on the challenge of writing their own program to get their Edison robot through a maze.

Grade levels	Difficulty	Prerequisites	Duration
Year 6+	Beginner	None[6]	55 minutes

Computer science and computational thinking topics	□ Sequence □ Introduction to functions □ Introduction to inputs, input parameters and outputs
Tie-ins to other subjects	□ n/a

Supplies you need

- Full set of Edison robots[7] and EdComm programming cables
- Full set of prepared programming devices (computers)
- 4x AAA batteries per robot
- Print or digital copies of the student sheets and print-outs of the activity sheet (maze)

Some great advice from the Hour of Code team

It's okay not to know! Respond to student questions and struggles with phrases like:

- "I don't know. Let's figure this out together."
- "Technology doesn't always work out the way we want."
- "Learning to program is like learning a new language; you won't be fluent right away."

And don't forget to have fun! (^_^)

[5] https://meetedison.com/robot-programming-software/edpy/

[6] You and your students should be familiar with Edison and EdPy already, however. The *Getting started with Edison and EdPy* set available at https://meetedison.com/robotics-lesson-plans/get-edison-moving-in-edpy/ has a step-by-step activity for introducing Edison and EdPy. If this is your first-time using Edison or EdPy, start with that guide and activity.

[7] This activity assumes Version 2.0 Edison robots. If you have Version 1 Edison robots, you will need to ensure students select Version 1 robots when loading EdPy and only use time as the constant for distance units. See more information in The *Getting started with Edison and EdPy* set available at https://meetedison.com/robotics-lesson-plans/get-edison-moving-in-edpy/

Part 1: How do computers 'think'?

Overview

The concept of how computers get and follow instructions, including the importance of sequence, is investigated. Students write two basic programs; the 'drive forward' and 'drive forward, then backwards' programs; learning how to code in a text-based environment. Writing and running the 'drive' programs also gives students a chance to see sequence in action while being introduced to inputs, input parameters, outputs, and functions.

Delivery recommendations

- Recommended time: 20 minutes
- If you are using Version 1 Edison robots with EdPy, you need to make sure that students always have the correct setup code[8]. Have students select Version 1 when they launch the EdPy app to automatically change the setup code in all new programs. Pre-made programs, including some of the example programs shown in this activity, will need to have the following changes made to the setup code[9]:
    1. Ensure that the Ed.EdisonVersion = Ed.V1
    2. Ensure that the Ed.DistanceUnits = Ed.TIME

Tips and tricks

- Students can have up to 5 programming tabs open at a time, which can be helpful if they want to refer back to other programs as the work.
- To help students understand input parameters (and how they effect outputs) have the students run the 'drive' programs a second time. Ask students to try using different input parameters in their 'drive forward' and 'drive forward, then backwards' programs and run the modified programs in their robot. Discuss how changing inputs effects the robot's behaviour (outputs).

[8] Edison V1 robots can only use time, not CM or inches, as the input parameter for distance units. That's why the setup code must always be Ed.DistanceUnits = Ed.TIME. The time distance unit is milliseconds, which means that to drive for 2 seconds, you need to set the distance input parameter to 2000 (since 2000 milliseconds = 2 seconds).

[9] If a program was made using Ed.CM as the distance units constant and you change it to Ed.TIME, you may also need to change the input parameter's values inside the program to adjust for this change. Why? If the original program said to drive for '5' where '5' was CM and it is now milliseconds, the result will be very different!

Part 2: How do you navigate a maze?

Overview

It's time to experiment! Students apply what they've learned about sequential coding using the 'Ed.Drive()' function with different input parameter values to write a program that will get their Edison robot through a maze.

Delivery recommendations

- Recommended time: 35 minutes
- An example programming solution can be seen at http://www.edpyapp.com/share/I2AJs
- Want even more bonus challenges? An additional extension is to have students make mazes to exchange with each other, solving the maze their classmate has designed.

Tips and tricks

- If students are feeling stuck, suggest they first plan out what actions the robot will need to take to get through the maze 'offline'. They can use their plan to help write their program.
- Remind students they can read through the documentation about the 'Ed.Drive()' function for more information about using that function. Reading documentation and being able to extract from it what you need to write code is an important skill in general-purpose programming.
- When turning, if you have Ed.DistanceUnits set to be either Ed.CM or Ed.INCH, the robot will convert the value you enter as the distance unit parameter to degrees.
  - If you have Ed.DistanceUnits = Ed.TIME the distance unit parameter will be time in milliseconds.
- Due to minor mechanical differences in the motors and encoders inside different Edison robots, some robots may not turn to exactly the value of degrees input. For example, the robot may not turn exactly 90 degrees when given the input of 90. Encourage students to try different values around the value they want (e.g. 88 or 93 if they want the robot to turn 90 degrees) to find the input that works best for their Edison.
- First attempts in programming are almost always 'wrong' – and that is perfectly okay! Persevering and changing programs iteratively, testing as they go, will get students to the right answer. Resilience is key!

What should I do if a student finishes the lesson early?

- Get students to complete all of the bonus challenges.
- Or, ask students who finish early to help classmates who are having trouble with the activity.