feat-ds

Electronics and Employability Advancement for Adults with Down Syndrome

2023-1-LV01-KA220-ADU-000160601

Funded by the European Union

WP3.2 TRAINING FORMAT

The Training Format is based on tinkering methodology applied to electronics and provides the intended audience with an engaging learning environment. Learners are able to learn at their own pace while accumulating step-by-step specific competencies, information and skills according to the chosen approach.

EFEKTAS GROUP

NGO NEST Berlin

Quality Culture

Electronics and Employability Advancement for Adults with Down Syndrome

2023-1-LV01-KA220-ADU-000160601

ELECTRONICS AND EMPLOYABILITY ADVANCEMENT FOR ADULTS WITH DOWN SYNDROME FEAT-DS

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or State Education Development Agency (SEDA). Neither the European Union nor the granting authority (SEDA) can be held responsible for them.

Funded by the European Union

Executive Summary

This Training Format has been developed within the framework of **FEAT-DS – Facilitating Electronics, Accessibility and Tinkering for Down Syndrome Adults**, a European project aimed at improving access to technological learning for adults with Down syndrome, which focuses on producing accessible, high-quality educational resources that integrate tinkering practices and introductory coding activities. The Training Format provides a structured and adaptable learning pathway introducing fundamental concepts in electronics, basic circuit logic and simple coding. Through step-by-step activities, guided exercises and interactive games available on the project's Web Platform, learners can develop practical competences at their own pace. The overall objective is to offer tools that support experiential learning, enhance autonomy, and foster greater participation in educational and employment opportunities for adults with Down syndrome.

The **FEAT-DS** project addresses the persistent barriers that adults with Down syndrome face in accessing technical and digital education. A transnational research phase conducted by the partners gathered insights into the educational needs of the target group and reviewed existing training opportunities in each participating country. These findings provided the foundation for designing training materials that are relevant, realistic and aligned with the identified needs. Adult educators and electronics specialists jointly contributed to the co-design process, enabling the integration of pedagogical expertise with technical knowledge. This synergy supports the creation of materials that are both accessible and technically sound, and that can be meaningfully used by educators working with adults with Down syndrome. The Training Format presented here includes:

- Thematic modules developed through a gradual, step-by-step approach;
- Activities combining basic electronics, logical reasoning and introductory coding tasks;
- Clear instructions to support implementation and replication;
- Direct links to the interactive coding games hosted on the FEAT-DS Web Platform.

Funded by the European Union

TABLE OF CONTENTS

Funded by the European Union

MODULE 1: HANDS-ON EXPLORATION: BUILDING WITH SIGHT AND TOUCH

Funded by
the European Union

Overview

This module is called “Hands-On Exploration: Building with Sight and Touch”. It is all about learning by doing. The best way to understand electronics is to touch the parts, see how they connect and watch what happens. This hands-on way of learning is called “tinkering”. Tinkering means you get to play with tools and materials to see how they work. There are no mistakes in tinkering, only new discoveries!

We will start by looking at some basic electronic parts. You will learn what they are called and what they do. Then, you will use these parts to build simple projects, called circuits. You will make a light turn on, a buzzer make a sound and a motor spin. You will see right away how your actions make something happen. This helps us understand how everything works together.

Later in the module, we will also try some simple coding. Coding is like giving instructions to a computer. We will use big, colorful blocks on the screen that are easy to use. You will write a simple code to make a light blink on and off. This will show you how instructions on a computer can control things in the real world. This whole module is designed to be fun, visual and a great first step into electronics.

Aims

- Learn that the best way to understand new things is by doing, seeing and touching them.
- Get to know the names and jobs of some basic electronic parts.
- Understand the simple idea of an electric circuit.
- Feel confident and successful by building things that really work.
- Discover how simple coding can control electronic parts.

Funded by
the European Union

Expected Outcomes

- You will be able to look at a battery, an LED light and a switch and say their names.
- You will be able to explain that a circuit needs to be a complete path for electricity to flow.
- You will be able to connect a battery and a switch to make a light turn on and off.
- You will be able to build a simple circuit that makes a buzzer create a sound.
- You will be able to use simple code blocks on a computer to make a real LED light blink.

Funded by
the European Union

1. THE BIG IDEA - LEARNING WITH YOUR HANDS

Learning by Doing

Have you ever learned how to tie your shoes? Or how to make a sandwich? You probably learned by watching someone and then trying it yourself. You used your hands, your eyes and you practiced.

That is the best way to learn new things! We call it **learning by doing** or **experiential learning**.

In this module, we will learn about electronics in the same way. We will not just read about it. We will build things. We will touch the parts. We will see what happens when we connect them. This is a powerful way to learn because you can see the results of your work right away. It is fun and helps you remember what you learned.

What is a Circuit?

Everything we build in this module will be a circuit. A circuit is just a path for electricity to travel. Think about a racetrack. The race cars need to follow the track all the way around to finish the race. If there is a break in the track, the cars have to stop.

Electricity is like the race cars. It needs a complete path to travel from its starting point, do a job (like lighting up a light) and then go back to where it started.

A simple circuit has 3 main things:

1. **A Power Source.** This is where the energy comes from. For us, this will be a battery.
2. **A Path**. This is what the electricity travels along. For us, this will be wires.
3. **A Load.** This is the part that does a job. It could be a light, a buzzer or a motor.

When all these parts are connected in a full circle or loop, we have a complete circuit. And that is when the magic happens! The light will shine, the buzzer will buzz or the motor will spin. If there is any break in the path, the circuit is open and the electricity cannot flow

Funded by
the European Union

2. MEET YOUR TOOLS - A VISUAL GUIDE TO ELECTRONIC PARTS

In our activities, we will use a few common electronic parts. These parts are big and easy to handle. Let's meet them!

The Battery

Figure 1. Battery

This is a **BATTERY.**

Its job is to give power to our circuit. It is like the food that gives us energy to run and play.

Our battery has two sides. One is positive (+) and one is negative (-). The red wire usually connects to the positive side and the black wire connects to the negative side.

Funded by
the European Union

The LED

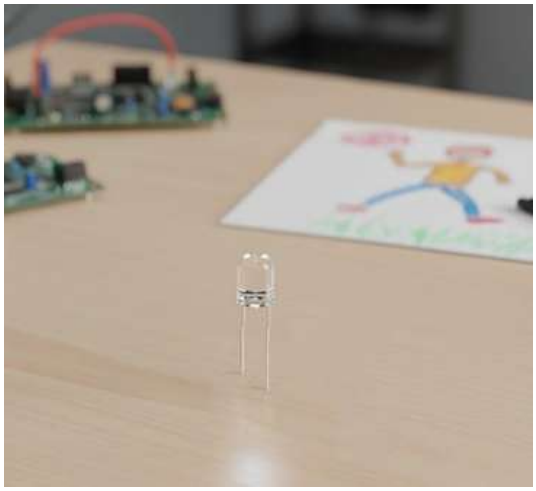


Figure 2. LED

This is an **LED**. LED stands for Light Emitting Diode. Its job is to **make light**. It is a small light bulb that shines brightly when electricity passes through it.

LEDs have two legs. One leg is a little longer than the other. This is important! The long leg must connect to the positive (+) side of the battery and the short leg must connect to the negative (-) side.

The Switch

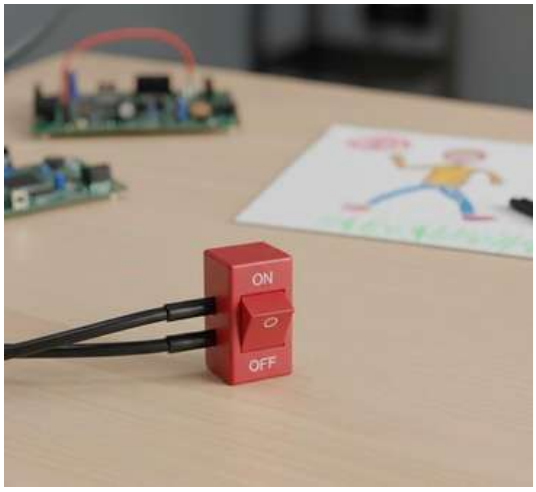


Figure 3. Switch

Its job is to open and close the circuit. It is like a gate on the racetrack.

When the switch is ON, the gate is closed and the path is complete. Electricity can flow.

When the switch is OFF, the gate is open and there is a break in the path. Electricity stops.

Funded by the European Union

The Buzzer

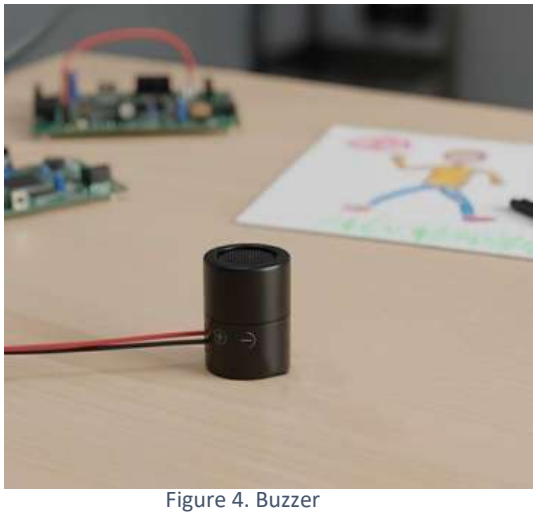


Figure 4. Buzzer

This is a **BUZZER**.

Its job is to **make a sound**. When electricity flows through it, it makes a buzzing noise.

Like the battery, the buzzer has a positive (+) side and a negative (-) side. We need to connect the wires the right way for it to work.

The Motor

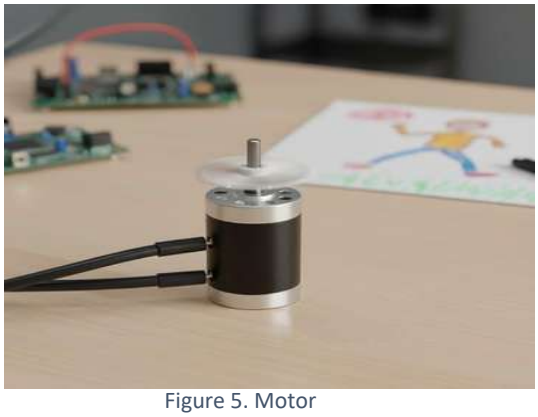


Figure 5. Motor

This is a **MOTOR**. Its job is to **spin around**. When electricity flows through it, a small shaft on top spins very fast. We can attach things like a fan or a wheel to it.

Funded by
the European Union

3. SAFETY AND SUPPORT - HOW WE WORK TOGETHER

Learning new things is an adventure. To make sure our adventure is safe and fun for everyone, we will follow a few simple rules together.

- **Work with a supervisor**. Your educator or helper is here to guide you. They can help you if you get stuck or have a question.
- **Use safe power**. We will only use low-voltage batteries (like AA batteries). We will never use power from the wall sockets.
- **Power off firs**t. Before you change any wires or parts in your circuit, always disconnect the battery first.
- **Ask for a check**. Before you connect the battery to a new circuit for the first time, always ask your supervisor to check it for you. This is the most important safety step.
- **Mistakes are okay!** Everyone makes mistakes when they are learning. It is a normal part of tinkering. If something does not work, it is not a problem. It is a chance to learn something new [2].
- **Celebrate your success**. When you get something to work, be proud! You are learning and creating. Positive feelings help us learn even better [3].

Funded by
the European Union

4. LET'S BUILD! - PRACTICAL TINKERING ACTIVITIES

Now it is time for the best part: building our own circuits! Remember, this is tinkering. Have fun, explore and see what you can create.

Activity 1: Make a Light Shine!

In this activity, we will build our very first circuit to turn on an LED light.

Goal for this activity:

- To build your first complete circuit.
- To see how electricity makes an LED light up.

What you will need (we will use large, color-coded parts that are easy to hold and connect):

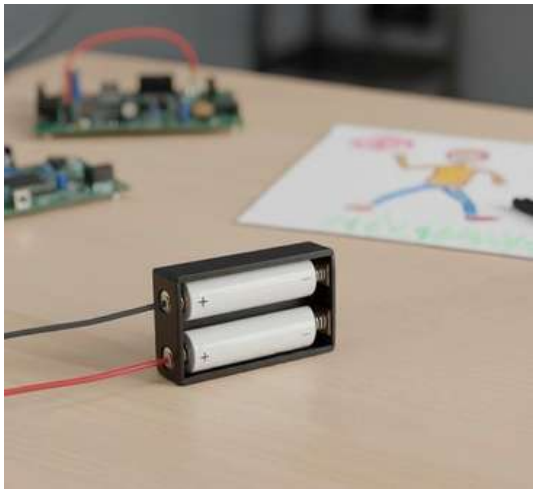


Figure 6. 1 Battery Holder with Batteries

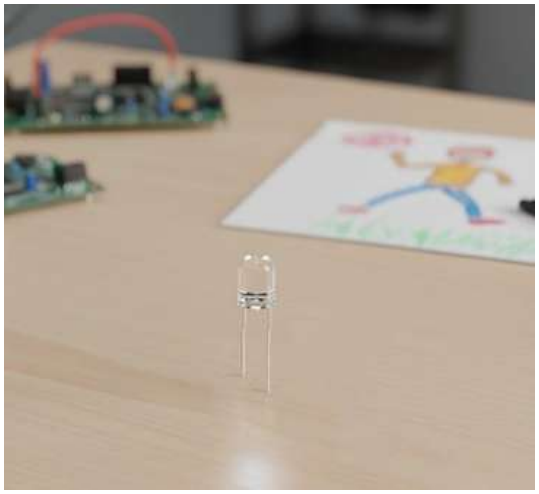


Figure 7. 1 LED (any color)

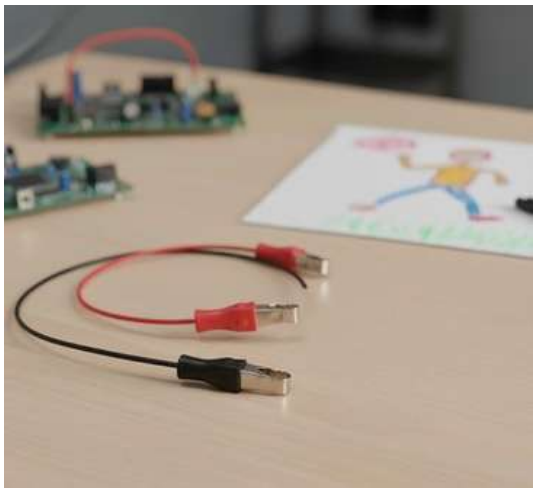


Figure 8. 2 Wires with clips (alligator clips)

Let's build it, step-by-step:

1. Get your battery.
Make sure the batteries are inside the holder. You will see a red wire (+) and a black wire (-) coming out.

2. Get your LED.
Look at the two legs. Find the longer leg. This is the positive (+) leg.

3. Connect the first wire.
Take one wire. Clip one end to the red wire from the battery. Clip the other end to the long leg of the LED.

4. Connect the second wire.
Take the second wire. Clip one end to the black wire from the battery.

5. Ask for a safety check.
Before the last step, ask your supervisor to look at your circuit to make sure it is correct.

6. Complete the circuit!
Now, clip the other end of the second wire to the short leg of the LED.

Educator's Note: If the LED doesn't light up, the most common reason is that it's connected backwards. Use this as a teaching moment. Guide the learner to disconnect the battery, reverse the LED, and try again. This reinforces the concept of polarity.

Funded by the European Union

Success!

Your LED should now be shining brightly! You have built a complete circuit. If it does not work, don't worry! Just check that all your clips are touching the metal parts and try again.

What did we learn?

We learned that for the LED to light up, electricity needs a complete path from the battery, through the LED, and back to the battery.

Challenge Yourself!

- Can you make the light shine without using the wires?
- Try using a different colored LED. Does it work the same way?

Activity 2: Make a Sound!

This time, we will make a circuit that creates a sound using a buzzer and a switch.

Goal for this activity:

- To use a switch to control a circuit.
- To build a circuit that makes a sound.

What you will need (we will use large, color-coded parts that are easy to hold and connect):

Funded by the European Union

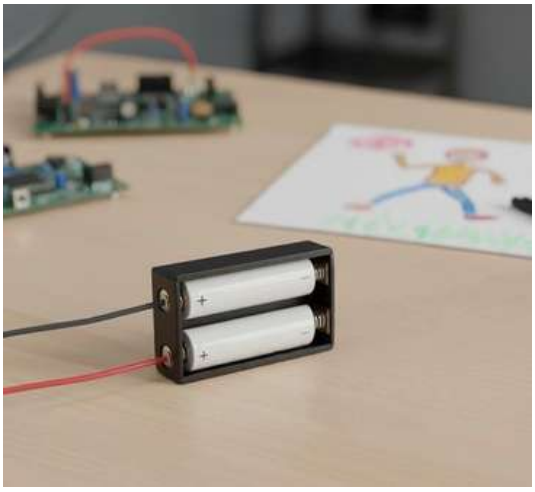


Figure 9. 1 Battery Holder with Batteries

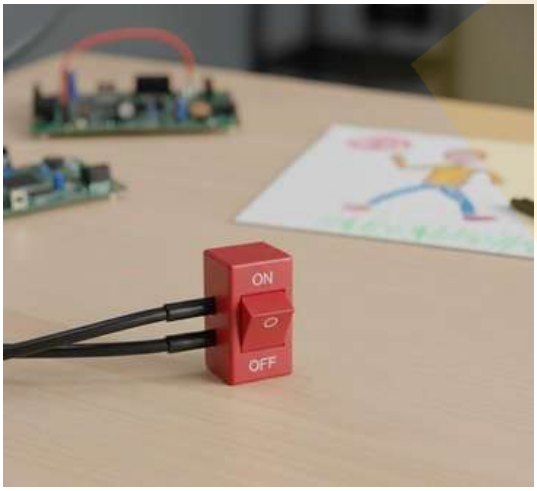


Figure 11. 1 Switch

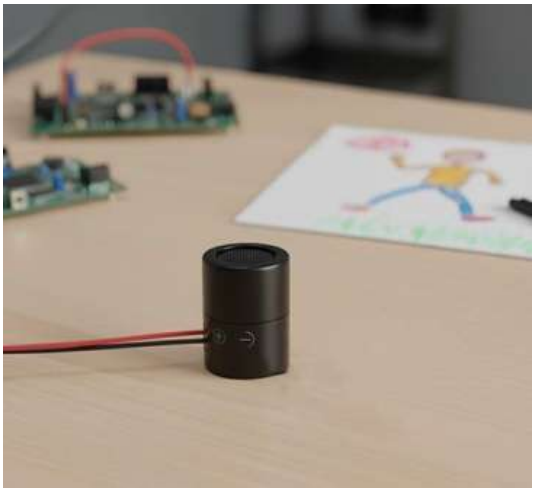


Figure 10. 1 Buzzer

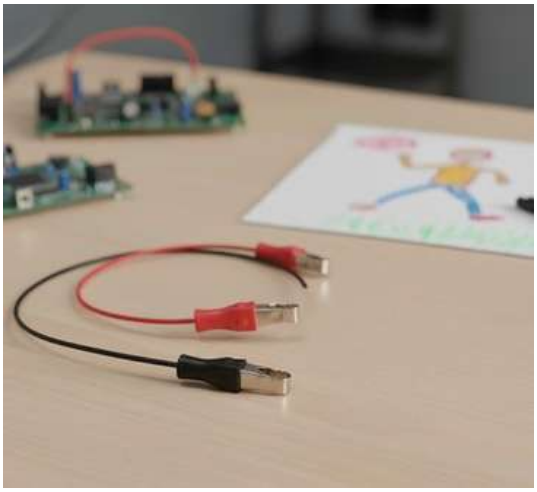


Figure 12. 3 Wires with clip

1. Connect the battery to the switch.
Take one wire. Clip one end to the red wire from the battery. Clip the other end to one of the metal legs on the switch.

2. Connect the switch to the buzzer.
Take a second wire. Clip one end to the other metal leg on the switch. Clip the other end to the red wire of the buzzer.

3. Connect the buzzer back to the battery.
Take the third wire. Clip one end to the black wire of the buzzer. Clip the other end to the black wire of the battery.

4. Ask for a safety check.
Ask your supervisor to look at your circuit before you turn it on.

5. Flick the switch!
Now, press or flick the switch to the ON position.

Funded by
the European Union

Success!

BZZZZZZ! The buzzer is making a sound. You used a switch to control the circuit. When you turn the switch ON, the circuit is complete. When you turn it OFF, the sound stops.

What did we learn?

We learned that a switch acts like a gate. It can open or close the path for electricity.

Challenge Yourself!

- Can you replace the buzzer with the LED from the first activity?
- Does the switch still work to turn the light on and off?

Activity 3: Make Something Spin!

For our last tinkering activity, let's make something move! We will use a motor to spin a small fan.

Goal for this activity:

- To build a circuit that makes something move.
- To see how electricity can be turned into motion.

What you will need (we will use large, color-coded parts that are easy to hold and connect):

Funded by the European Union

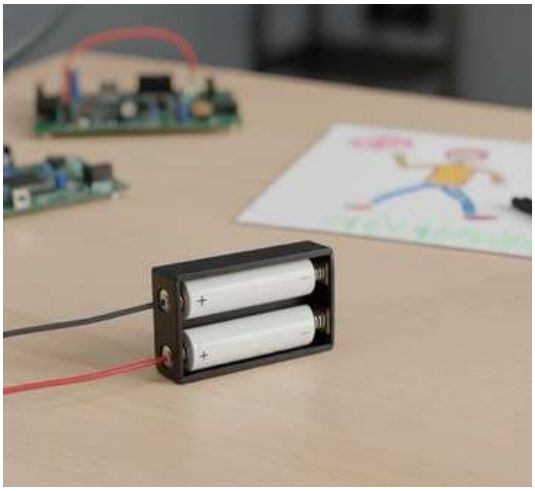


Figure 13. 1 Battery Holder with Batteries

Figure 14. 1 Motor with a fan attached

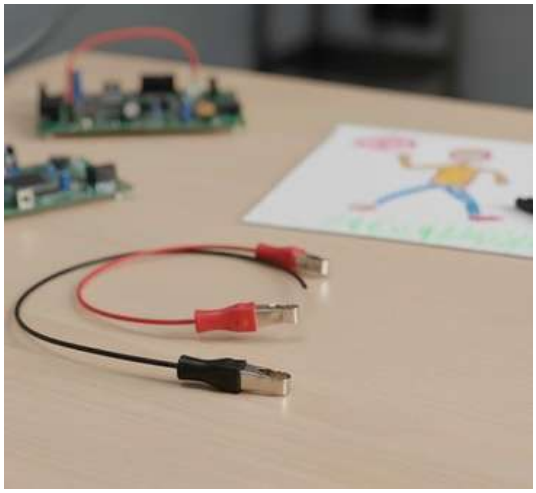


Figure 15. 2 Wires with clips

Let's build it, step-by-step:

1. Look at the motor.
You will see two small metal tabs on the back of the motor. This is where we will connect our wires.

2. Connect the first wire.
Take one wire. Clip one end to the red wire from the battery. Clip the other end to one of the metal tabs on the motor.

3. Connect the second wire.
Take the second wire. Clip one end to the black wire from the battery.

4. Connect the second wire.
Take the second wire. Clip one end to the black wire from the battery.

5. Ask for a safety check.
Before the last step, ask your supervisor to look at your circuit to make sure it is correct.

6. Complete the circuit!
Now, clip the other end of the second wire to the short leg of the LED.

Success!

The fan is spinning! You have built a circuit that turns electrical energy into movement energy. If it doesn't spin, check that your clips have a good connection and try again.

What did we learn?

We learned that electricity can do more than make light and sound. It can also make things move!

Challenge Yourself!

Disconnect the battery. Swap the red and black wires on the motor tabs.

- What happens to the fan when you connect the battery again?
- Does it spin in a different direction?

Funded by the European Union

5. TELL THE COMPUTER WHAT TO DO! - A SIMPLE CODING ACTIVITY

You have learned how to build circuits with your hands. Now, let's learn how to control a circuit with a computer. This is called **coding**.

What is Block Coding?

We will use a special kind of coding called **block coding**. Instead of typing words, you use colorful blocks that look like puzzle pieces. Each block is an instruction for the computer. You snap the blocks together to create a set of instructions, called a program [4].

Activity 4: Code a Blinking Light!

In this activity, we will write a simple program to make an LED light blink on and off all by itself.

Goal for this activity:

- To write your first computer program using code blocks.
- To see how code can control a real electronic part.

What you will need:

Funded by the European Union

Figure 16. A computer with the coding software open

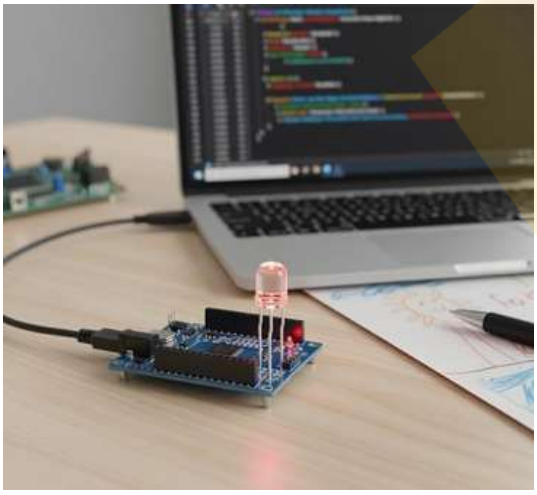


Figure 17. An LED light connected to the board

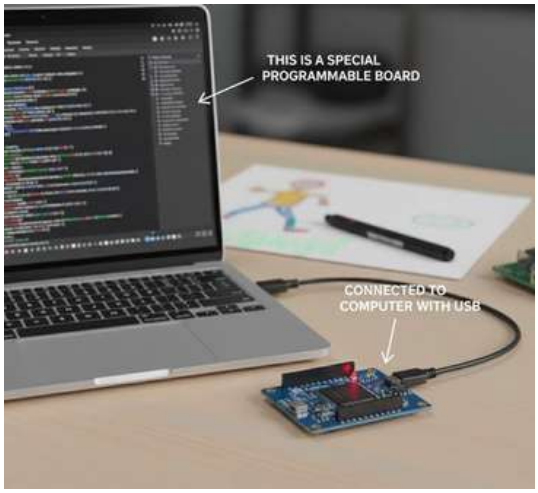


Figure 18. A special board (like an Arduino or Micro:bit) connected to the computer with a USB cable

Your educator will help you get this set up. **The setup should look something like this**:

1. Find the "turn light ON" block.
Look in the list of code blocks. Find the one that tells the light to turn ON.

2. Drag the block to your workspace.
Click and drag the "turn light ON" block into the empty program area.

Funded by
the European Union

3. Find the "wait" block.
Now find a block that tells the computer to wait. We will tell it to wait for 1 second. Drag it under the first block until it snaps into place.

4. Find the "turn light OFF" block.
Next, find the block that tells the light to turn OFF. Drag it and snap it below the "wait" block.

5. Add one more "wait" block.
We need the light to stay off for a moment. Add another "wait 1 second" block to the very bottom of your program.

6. Run your program!
Click the "Run" or "Start" button on the screen. Look at your real LED light on the board.

Educator's Note: This step connects the digital world to the physical world. Emphasize this connection by asking, "See how the blocks you put on the screen are telling the real light what to do?"

Success!

Your LED is blinking! It turns on for one second, then off for one second, over and over again. You wrote a program! You gave instructions to the computer and the computer is controlling a real object.

What did we learn?

We learned that code is a set of instructions. We can use block coding to tell electronics what to do and when to do it.

Challenge Yourself!

- Can you change the numbers in the “wait” blocks? Try making the light blink faster or slower.

Funded by the European Union

6. LET'S PLAY A GAME! - CODING GAME DESCRIPTION

To practice what we have learned about circuits, you can play a fun game on the computer or a tablet. Here is how the game works.

Game Title: Complete the Circuit!

Learning Goal: To learn that a circuit needs a complete path to work.

Game Concept: The game shows a battery on one side of the screen and a light bulb on the other side. In between, there are gaps in the path. Pieces of wire and a switch appear at the bottom of the screen. The player's job is to drag the pieces from the bottom and drop them into the gaps to complete the circuit.

How to Play:

- Look at the circuit on the screen. You will see where the path is broken.
- Look at the pieces at the bottom of the screen (wires, switch).
- Use your finger or the mouse to drag a piece up to a gap.
- Let go to drop the piece into the gap.
- Keep doing this until all the gaps are filled and the path is complete.
- Finally, click on the switch to turn it ON.

Feedback: When you successfully complete the circuit and click the switch, you get fun, positive feedback! The light bulb on the screen will light up with bright yellow rays. A happy, cheerful sound will play, like a "ding!" or a short, happy tune. To make it even more exciting, if a real electronics board is connected to the computer, the real LED on the board will light up at the same time as the one on the screen! This shows how the game connects to the real world.

7. CONCLUSION

Congratulations! You have completed Module 1.

Look at everything you have accomplished. You learned what a circuit is by building them with your own hands. You made a light shine, a buzzer make noise and a motor spin. You even became a coder and wrote a program to control a light.

You have taken the first big step into the amazing world of electronics. You have shown that you are a builder, a thinker and a creator. You should be very proud of yourself.

We hope you had a lot of fun tinkering and exploring. Keep being curious, keep asking questions and keep building. We are excited to see you in the next module for even more fun with electronics!

Funded by
the European Union

8. FURTHER READING

If you had fun and want to explore more, here are some great places to look online. You can ask your educator to help you find them.

- **Tinkercad Circuits:** https://www.tinkercad.com/circuits

This is a free website where you can build and test circuits on the computer. It is like a digital version of the activities we did and it is very colorful and easy to use.

- **Scratch:** https://scratch.mit.edu

This is a very popular website for block coding. You can make stories, games and animations by snapping blocks together. It is a great way to practice your coding skills in a fun, visual way.

- **Code.org - Hour of Code Activities:** https://code.org/hourofcode/overview

This website has many fun, one-hour coding games and tutorials. Many of them use characters you might know from movies and games. They are great for beginners.

- **YouTube Channel: Simple Electronics:** https://www.youtube.com/c/SimpleElectronics

This channel has many videos that show how to build simple and fun electronics projects. The videos are clear and show each step, which is great for visual learners.

- **Assistive Technology Internet Modules (ATIM):** https://atinternetmodules.org

This website is a great resource for educators and families. It has free, self-paced modules with videos and case studies that help users learn more about assistive and supportive technologies for individuals with disabilities.

Funded by the European Union

9. REFERENCE LIST

- https://lvp.digitalpromiseglobal.org/content-area/adult-learner/strategies/experiential-learning-adult-learner/summary
- https://www.down-syndrome.org/en-gb/library/news-update/06/1/inclusive-education-individuals-down-syndrome/
- https://www.positiveaction.net/blog/teaching-students-with-down-syndrome-strategies
- https://codakid.com/block-coding/
- https://www.tinkercad.com/circuits
- https://scratch.mit.edu
- https://code.org/hourofcode/overview
- https://www.youtube.com/c/SimpleElectronics
- https://atinternetmodules.org

Funded by the European Union

10. SELF-ASSESSMENT GRID

Let's see what you remember! Choose the best answer for each question. The correct answer is in **bold**.

Nr.	Question	A	B	C	D
1	What is the job of a battery in a circuit?	To make light	**To give power**	To make sound	To spin
2	What do we call a complete path for electricity to flow?	A line	A box	**A circuit**	A street
3	Which part makes a buzzing sound when electricity flows through it?	An LED	A motor	A switch	**A buzzer**
4	An LED has two legs. Which leg should connect to the positive (+) red wire?	The short leg	**The long leg**	Either leg	The bent leg
5	What does a switch do?	Makes things brighter	Makes things louder	**Opens and closes the circuit**	Gives the circuit more power
6	What is "tinkering"?	Reading a book	Listening to a teacher	**Learning by building and exploring**	Watching a video
7	Which of these parts spins around when you make a circuit with it?	A battery	An LED	A buzzer	**A motor**
8	In block coding, what do you use to give instructions?	Typing words	Drawing pictures	**Colorful blocks**	Speaking out loud
9	What is the most important thing to do before you turn the switch on for the first time?	Make the room dark	Tell a friend	**Ask a supervisor to check your circuit**	Shake the battery
10	In our coding activity, what did our program make the LED do?	Stay on	Stay off	Get brighter	**Blink on and off**

Funded by the European Union

MODULE 2: CLEAR COMMUNICATION IN ELECTRONICS: SPEAKING THE LANGUAGE OF CIRCUITS AND CODE

Funded by
the European Union

Overview

This module is entirely dedicated to a fundamental concept: clear and accessible communication is the key to learning electronics and programming. For adults with Down syndrome, having simplified instructions, visual supports, and well-defined steps is not just helpful—it is essential for mastering these technical skills.

Research conducted across several countries has shown that challenges such as verbal memory and abstract thinking can make learning more difficult. That's why our approach focuses precisely on these aspects. By using simplified language, supported by pictograms and visual diagrams, we help participants learn how to decode electronic instructions and understand the basic logic of programming.

Aims

- To develop ability to interpret simplified technical instructions in electronics and programming contexts.
- To build recognition and understanding of standard electronic symbols, pictograms, and visual diagrams.
- To establish foundational skills in logical sequencing and step-by-step thinking for technical tasks.
- To introduce block-based programming as an accessible entry point into coding.
- To foster self-advocacy skills for recognizing when clarification is needed and asking effective questions.
- To build technical confidence through structured, successful experiences with electronics and coding.
- To develop ability to interpret simplified technical instructions in electronics and programming contexts.

Funded by the European Union

Expected Outcomes

- Participants will read and interpret technical instructions written in simplified language formats.
- Participants will recognize and name at least ten common electronic components and their schematic symbols.
- Participants will follow multi-step visual guides to assemble simple circuits independently.
- Participants will understand basic block-based programming structure and create simple programs with 5-10 blocks.
- Participants will demonstrate systematic troubleshooting skills when projects don't work as expected.
- Participants will exhibit increased confidence in approaching technical tasks and persisting through challenges.
- Participants will develop effective communication strategies for asking technical questions and requesting help.
- Participants will apply clear communication principles in their own work, including organizing materials and checking progress.

Funded by
the European Union

1. WHY CLEAR COMMUNICATION MATTERS

How Accessible Language Opens the Door to Technical Learning

In our everyday life, we are used to receiving instructions, whether it is to follow a recipe, assemble a piece of furniture or simply use a device. For the task to be enjoyable, it is important that the instructions are simple and intuitive. The more complex, confusing or unclear they are, the more frustrated and demotivated we feel, with the risk of not being able to finish what we have started. This reasoning is even more true in the field of programming or electronics, where maximum precision is required because even what may seem like a small error can trigger a chain reaction and prevent a circuit from functioning properly or a programme from running.

Figure 1. Two people communicating

As research in various countries consistently documents, for adults with Down syndrome, the importance of interpreting instructions is amplified by specific cognitive characteristics. The transnational analysis conducted by the partners of the FEAT-DS project in Lithuania, Latvia, Italy, Germany and Bulgaria identified reduced short-term verbal memory as a common difficulty among adults with Down syndrome.

While the research highlighted weaknesses in learning, such as long and complex sentences, it also highlighted strengths that should be capitalised on: visual-spatial processing is often a well-developed skill in people with Down syndrome, which means that they understand and remember information better when it is presented in visual form rather than exclusively verbally. The combination of simplified language, visual support and tactile involvement enables people with Down syndrome to acquire technical skills that would otherwise be beyond their reach.

Example:

A traditional technical manual might say:
"Before beginning the assembly procedure, ensure that all components have been identified and systematically organised according to their functional categorisation."

This sentence, while technically correct, presents many barriers: it uses formal and academic vocabulary, employs the passive voice, introduces abstract concepts such as "functional categorisation" and encompasses multiple ideas in a single complex syntactic structure. A person with Down syndrome reading this instruction might struggle to grasp its essential meaning, feeling overwhelmed before even beginning the task.

A clear rephrasing could be:
"Before you start building, look at all your pieces. Put each piece in its place on the table. This will help you find what you need."

This sentence is completely different in that it uses clear, everyday language and breaks the information down into three short, concise sentences. Furthermore, it not only explains what to do but also the purpose of why you need to do these two actions. This approach ensures that the organisational principle will also be used in future projects.

Funded by
the European Union

2. WHY CLEAR AND SIMPLE INSTRUCTIONS ARE VITAL FOR SUCCESS

Every day, we use instructions to do things: they tell us how to cook, how to travel, or how to build something new. Think about instructions like a small map: they guide us step by step until we reach our goal. And when the map is clear, we can follow it easily! But when the map is messy or confusing, we can get lost. The same happens when we learn new skills especially when we work with electronics or computers. There are often many parts, tools, and actions to remember. Sometimes it gets so confusing!

If the instructions are too long or too complicated, we may not know what to do next. This can make us feel unsure, tired, or frustrated. But **when the instructions are short, simple, and well-organized, everything becomes easy to understand.**

But what makes an instruction, a clear one?
A clear instruction explains to us what to do, but also why we do it. For example, instead of saying “Connect all the wires”, a clear instruction should say “Connect each wire carefully, so the circuit will work”. As you can see, this instruction tell us:

1. What to do: connect all the wires.
2. Why we do it: to make the circuit work.

This way, we remember better and feel more motivated and aware of what’s happening.

Educators’ note:
Instead of saying “Before you start building, make sure you have all the components available and in the right order,” we can say, “Look at all your pieces. Put each one on the table. This helps you see what you have.” Both versions give the same information, but the second one is easier to follow and less stressful.

Funded by the European Union

3. LEARNING TOGETHER AND UNDERSTANDING THROUGH VISUALS

When we learn something we use also our eyes, not only the brain! In fact, our eyes help our brain understand and remember information faster. This is why visual aids (like pictures, symbols, colours, and diagrams) are so important in learning electronics and coding.

In electronics, a **picture** can show how the pieces fit together. You can see where to place the battery, how to connect the wires, or which direction a component should face. A **symbol** can remind you what each part does: a small light bulb for "LED," a zigzag line for "resistor," a plus sign for "positive." These small signs are like a universal language: you can understand them even without words. In coding, **visual blocks** show the logic of the program. Each block has a colour, a shape, and a place in the sequence. You can see how actions connect: one block starts a movement, another repeats it, another checks a condition. By looking at the blocks, you understand how the computer "thinks."

Visuals also make **learning more inclusive!**

Sometimes words are difficult to read or remember: a clear picture or a coloured diagram can make the same information easier to understand and helps you see the whole picture before you start, following each step confidently.

For example, before you begin an assembly or coding activity, you might look at a drawing that shows all the materials. You can then place each piece in the same position on your table. When you know what each part looks like and where it goes it is much easier to work calmly and avoid mistakes.

Finally visuals make learning enjoyable! They bring colour and movement to the process, and you can follow the steps like a story, from the first image to the final result.

This keeps your attention and helps you remember what you have learned.

Funded by the European Union

ACTIVITY 1: BUILD A SIMPLE FLASHLIGHT

Now it is time to practise!
You will build a small electronic object by following a step-by-step visual guide. The goal is to understand how clear words and clear pictures work together to make learning easier.

What you will need:

- 1 small LED light bulb
- 1 battery (AA or coin cell)
- 1 battery holder (if needed)
- 1 small switch (optional)
- 2 short wires
- Tape or small clips
- This visual guide (printed is better)

What you will learn:

- Practise following simple written and visual instructions.
- Understand the idea of a flow: step by step, one action leads to the next.
- Learn how visual aids make complex ideas easier to understand.
- Work with a partner to communicate clearly and solve small problems.
- Build confidence through a real and visible result.

Funded by
the European Union

Step-by-step instructions:

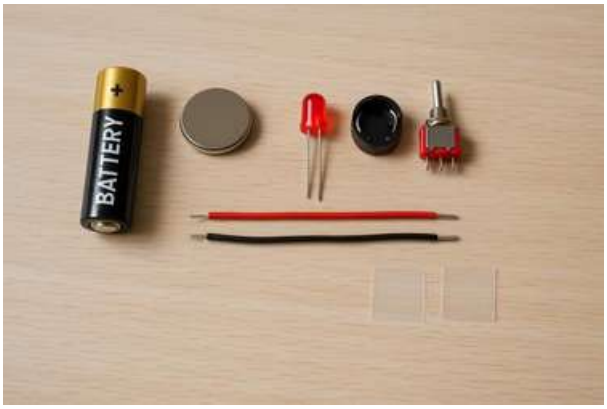


Figure 2. Materials needed

1. **Look at your materials.**

   Place all your pieces on the table.

   Make sure you have everything before you begin.

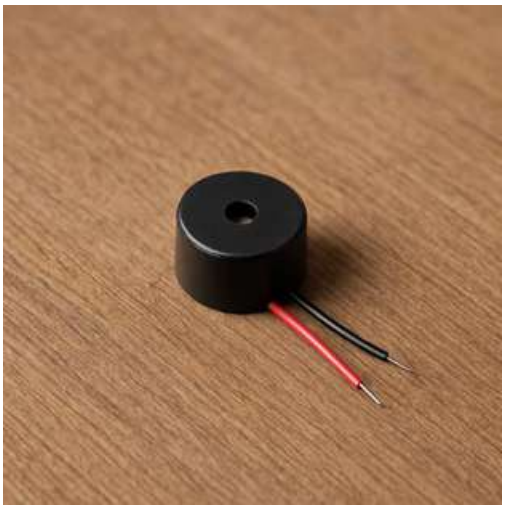


Figure 3. LED

2. **Find the battery and the LED.**

   Notice that the LED has two legs, one long and one short.

   The long leg is the positive side.

Funded by the European Union

Figure 4. LED and battery

3. **Connect the LED to the battery.**

Touch the long leg of the LED to the positive end of the battery.
Touch the short leg to the negative end.
You should see the light turn on.

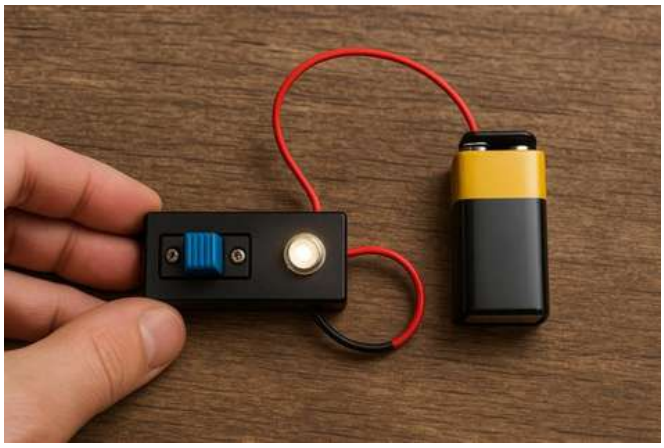


Figure 5. Connecting the switch

4. **If you have a switch:**

Place the switch between one of the connections, so you can open and close the circuit.
Try turning the light on and off.

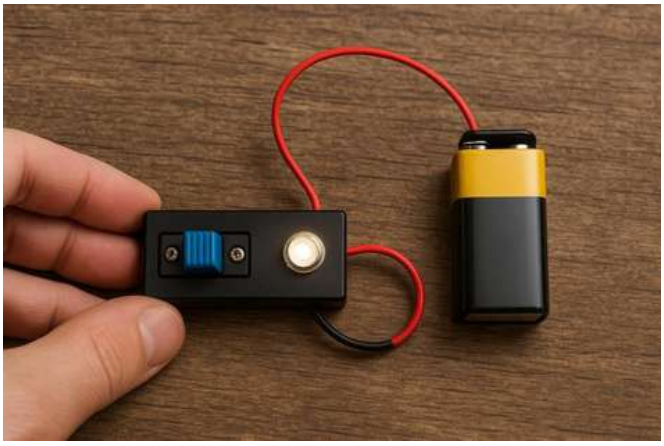


Figure 6. Securing the connection

5. **Secure the connections.**

   When everything works, use tape or clips to hold the wires and battery together.

Figure 7. Completed circuit

6. **Observe what happens.**

   When the circuit is closed, the light shines.
   When it is open, the light turns off.
   This shows how electricity moves through the parts.

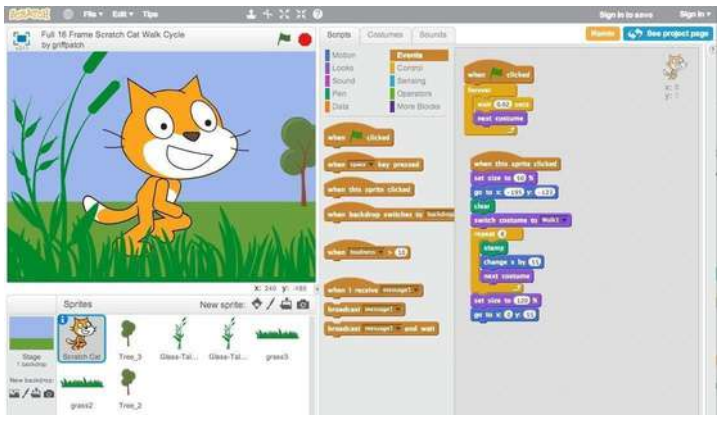
4. INTRODUCTION TO BLOCK-BASED PROGRAMMING LANGUAGE

Why should we use the block-based programming language?

Programming, or coding, means basically telling a computer what to do, explaining every step. Each instruction we give, is a part of a whole recipe: we carefully follow the order of the ingredients, and the computer does exactly what we ask it to do.

But why most people are scared of programming?
In **traditional programming**, these instructions are written as text, using words, numbers, and symbols. But even a very small mistake (such as a missing comma or an extra letter, exactly like a missing ingredient) can stop the program from working. For beginners, this can be confusing and sometimes frustrating.

But **block-based programming** makes this process much simpler and enjoyable!
Instead of writing long lines of complicated text, we use **colorful blocks** that connect together like pieces of a puzzle. Each block represents one action that we ask the computer to do, such as move “move the character”, “wait” and so on.
When we put the blocks together in the right order, all these blocks form a complete program that the computer can easily understand, showing us the amazing and interactive project we created.
This way we turn coding into something playful and intuitive, that allows learners to concentrate on the **logic of actions and results** rather than worrying about spelling mistakes.



The interface of Scratch, one ofthe main block-based programs

Funded by
the European Union

That's why block-based programming is designed for people who learn best through **seeing and doing**. It is especially useful for adults with Down syndrome also because:

- It uses colors and shapes to organize information.
- It gives immediate feedback: you can see right away if something works.
- It avoids confusing syntax errors (no typing mistakes).
- It allows learners to experiment freely without fear of breaking anything.
- It turns abstract concepts into something visible and concrete.

Let's understand the basics of block-based programming language

Before starting to code games and all the fun things that you can imagine, it is important to understand how a block-based programming environment works so we can better navigate the tool without fear or frustration.
This type of programming is designed to be visual, simple and fun. In fact, instead of writing long lines of text, learners build programs by putting together colorful blocks. Each one of these blocks are linked to a specific action or instruction.
The blocks fit together like pieces of a puzzle, to help you manage the logic and order of operations in a visual way. So, remember: if the pieces don't match, you have to think about another solution!
When we connect the blocks, we are actually building a story for the computer to follow: the computer reads the blocks from top to the bottom, just as we read a list of steps in a recipe.

Every block-based platform such as Scratch, MakeCode, or Code.org, organizes blocks into color-coded categories.
Each color represents a type of action, making it easier to find what we need.

Below are **the most common categories and what they do**:

Category of the block	Color of the block	What the block does
Event	Yellow	Starts the program (e.g. “When the green flag is clicked”)
Movement	Blue	Moves a character or object
Appearence	Purple	Changes color, shows or hides a message
Sound	Pink	Plays a sound or music
Control	Orange	Repeats or waits (“Repeat 5 times”, “Wait 2 seconds”)
Sensing	Light blue	Detects if something happens (e.g. “If button pressed”)
Variables	Dark Orange	Saves and uses information such as a number or score

Understanding How Programs Work

Every program we create with these blocks follows a clear sequence: this sequence is what gives computers their ability to do exactly what we want them to do.

But what is a sequence?
A **sequence** is, basically, the block that you connect together to create the program. The computer that plays the program reads the instructions written in the blocks, from top to bottom, just as when we read a page or follow a recipe!

This sequence is really special, because you can read the structure of your thinking, but also you can change your idea by moving a block higher or lower, or removing it completely.

Funded by the European Union

And these changes will have a visible and immediate effect! This will help you understand how one action influences another, teaching you an important skill: **thinking logically and predicting results**.

That's why we love this coding language so much: it is a way to train the mind to organise ideas, recognise patterns and build smart and creative solutions step by step.
All valuable skills for everyday life, not just computer time!

ACTIVITY 2: Catch The Mouse

What you will need:

- A computer
- Internet connection

What you will learn:

- Understand that a program follows a flow of instructions from top to bottom.
- Recognize how different blocks control actions and reactions.
- Use events, loops, and conditions to organize a simple sequence.
- See how connected instructions create interactive behaviour between sprites.
- Practise problem-solving and adjustment through testing.
- Build confidence and curiosity about how code works visually.

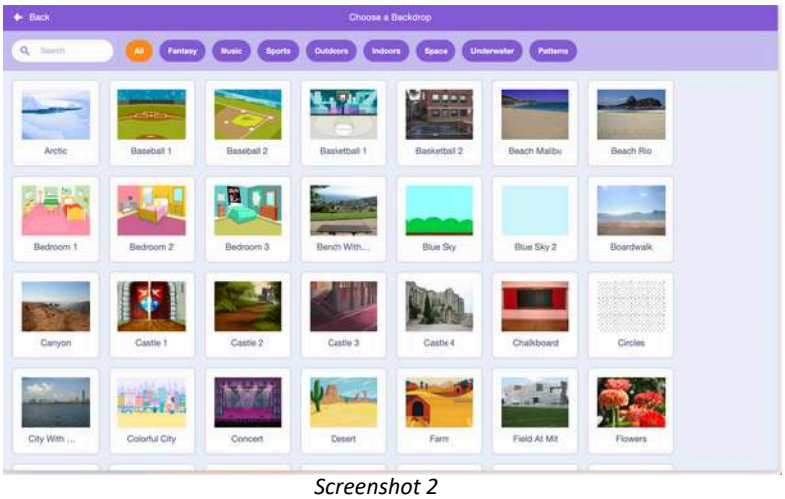
<u>Let's Code the Game!</u>

Screenshot 1

Create your project:

1. Open Scratch and click Create.
2. Delete the default sprite.
3. Add a new sprite for the cat.
4. Add another sprite for the mouse.

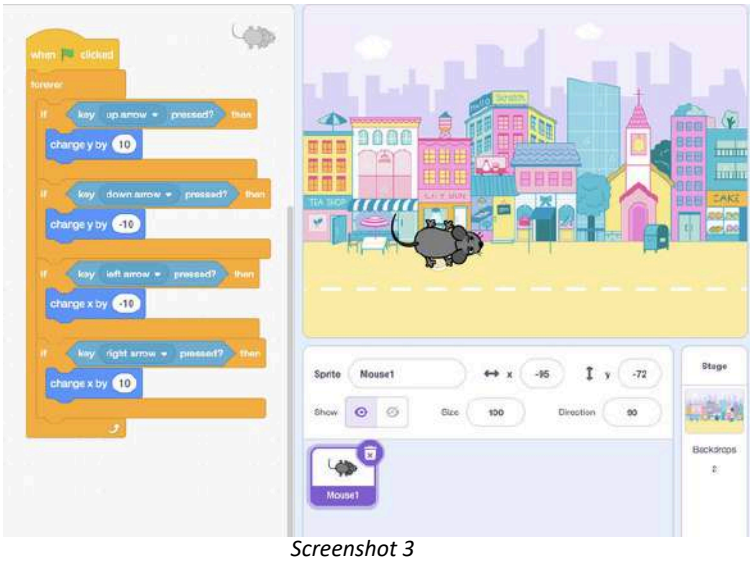
Funded by the European Union



Screenshot 2

Choose the background:

1. Click the Choose a Backdrop icon (bottom-right corner).
2. Select a nice background where your game will take place.
3. Go back to the coding area.



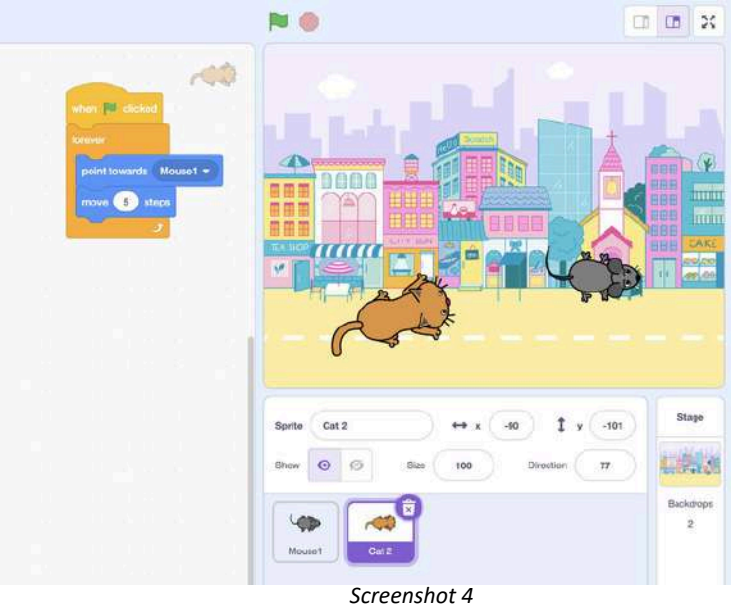
Screenshot 3

Make the mouse move:

1. Click on the **mouse** sprite.
2. Go to the **Events** category and drag the block: “When green flag clicked.”
3. Go to **Control** and drag a “forever” loop.
4. Inside the “forever” loop, add four “if” blocks from Control, one for each arrow key.

Funded by
the European Union

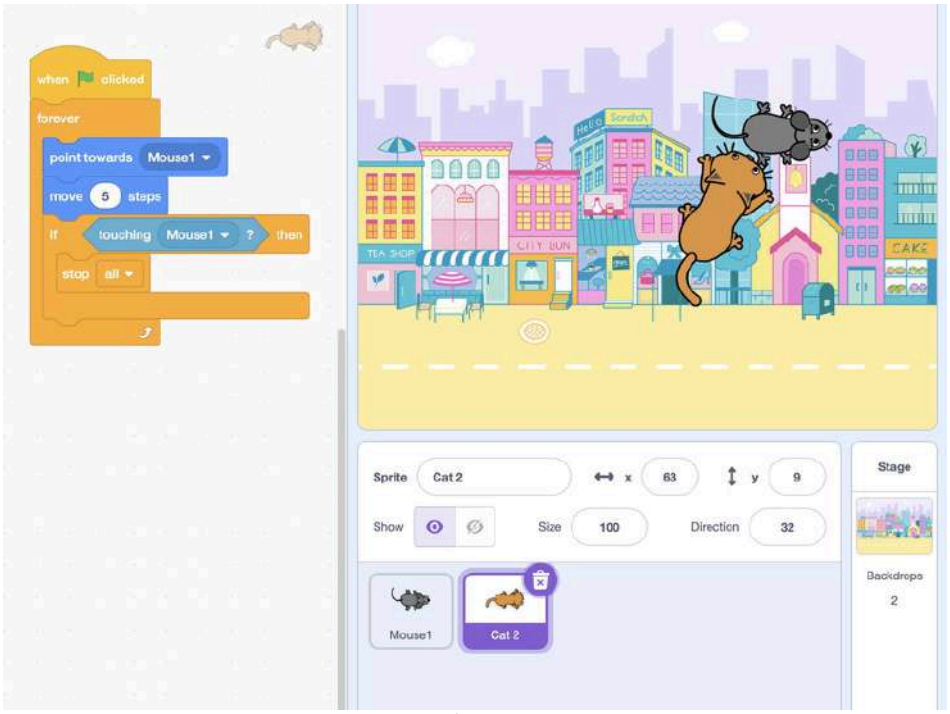
5. From **Sensing**, choose “key (arrow) pressed?” and put one in each “if” block.
6. From **Motion**, add these inside:
- For the up arrow: “change y by 10.”
- For the down arrow: “change y by -10.”
- For the right arrow: “change x by 10.”
- For the left arrow: “change x by -10.”
Now your mouse can move up, down, left, and right!



Screenshot 4

Make the cat chase the mouse:
1. Click on the cat sprite.
2. Add “When green flag clicked.”
3. Add “forever.”
4. Inside, put the motion blocks:
- “point towards (mouse)”
- “move 5 steps”
Now the cat will keep following the mouse everywhere on the screen.

Funded by
the European Union



Screenshot 5

Make the game end when the cat chases the mouse:
1. Stay on the cat sprite.
2. Inside the “forever” loop, add:
- “if touching (mouse) then” (Control block + Sensing Block)
- Inside that block, add “stop all” (Control block)
Now, when the cat touches the mouse, the game will stop!

Test the game:
1. Click the green flag to start.
2. Use the arrow keys to move the mouse.
3. Watch the cat chase it!
4. If the cat is too fast or too slow, change the number in “move 5 steps.”
5. Try again and have fun experimenting.

Funded by
the European Union

5. TROUBLESHOOTING AND PROBLEM-SOLVING WITH CLEAR COMMUNICATION

Strategies for Identifying and Solving Problems When Things Don't Work

Troubleshooting is a fundamental skill in electronics and programming, even though it is often underestimated in traditional learning.

But what is **troubleshooting**?
Troubleshooting is the ability to identify the problem, understand what might be causing it and work to try out different solutions that might solve it.
This ability is just as important as the ability to create a programme or circuit correctly: in fact, even professional programmers and engineers spend a lot of time troubleshooting, as this is a normal (and very important) part of their job.
It is not a synonymous of failure or incompetence, but an important part of the process.

Troubleshooting can present both a challenge and an opportunity:
CHALLENGE - If you find learning something more difficult if it's related to abstract thinking and memory, this can impact troubleshooting, which requires managing numerous possible solutions.

OPPORTUNITY - When problem-solving processes are clear and structured specifically to accommodate these difficulties, perhaps with purely visual solutions and concrete strategies, you can start solving problems effectively.
In this chapter, we will explore how to develop problem-solving strategies and how to deal with this crucial stage of programming.

Remember that problems are normal
One of the most important lessons in troubleshooting is understanding that unexpected results are part of electronics and programming. They are not signs of failure, lack of intelligence, or inadequacy. Even the most experienced professionals regularly encounter problems in their coding projects, requiring them to spend time figuring out where the error is and how to fix it. But the difference between a beginner and an advanced programmer is that the latter are not discouraged by obstacles because they have their own problem-solving strategy.

Funded by the European Union

Troubleshooting strategies

Effective troubleshooting must follow clear processes rather than random attempts. Even if random attempts may occasionally lead to solutions, they are inefficient, frustrating and they don't teach you much. **Systematic troubleshooting**, on the other hand, **examines potential causes in a logical order, first checking the most likely or most easily testable possibilities and then moving on to the less likely causes**.

Template to follow when asking for help:

"I am trying to [describe what the student wants to do].
However, [describe the current result].
I have already checked [list of what has been checked or tried].
Can you help me understand [insert question or area where assistance is needed]?'

Concrete example:
"I'm trying to make my LED flash. At the moment, the LED is not turning on at all. I have already checked that the battery is connected, that the switch is on, and that all connections are secure. Can you help me understand if my LED is connected correctly?"

This approach to asking for help shows that the student has engaged in systematic troubleshooting, has taken responsibility for solving what they can on their own, and has a specific need that they are articulating clearly. It also simplifies the teacher's work, as they immediately understand what the problem is, what has been tried, and where to focus their help, while also engaging the student in the process.

Practising this communication pattern can be very beneficial in the learning process!

Educators' note: Educators guide the process, but they are also your partners in exploration!
When a teacher shows that even they can make mistakes, and then calmly fix them, learners see that errors are normal and solvable. Peers also play a key role!
When learners observe one another's methods, they discover different ways to think and work. Sometimes, explaining a solution to a friend helps both people understand better.

Learning together makes the experience friendlier, safer, and more enjoyable.
Over time, collaboration develops patience, empathy, and self-advocacy, essential skills for independent living and future work.

Funded by the European Union

ACTIVITY 3: “Cheese Chase” to practise sequencing, simple variables, and systematic troubleshooting

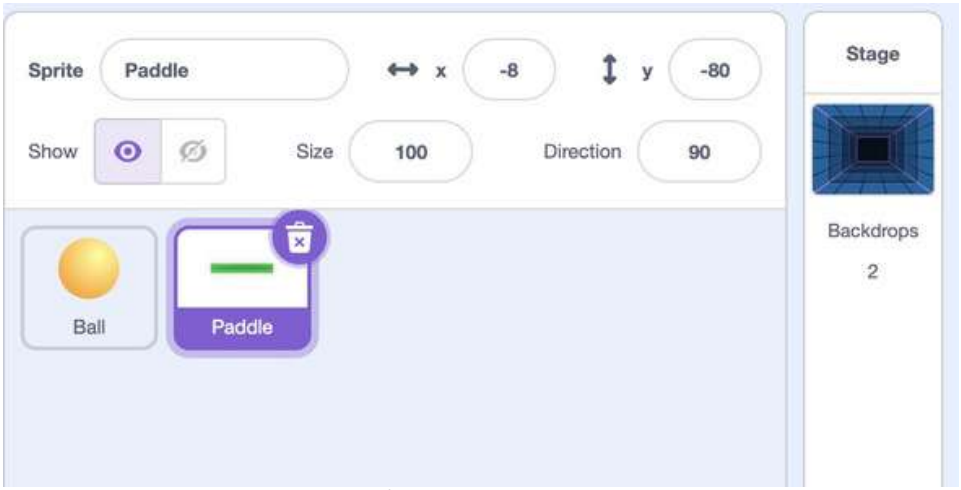
How the finished game works:
In this game you control a paddle. A ball moves around and must bounce off the paddle. Each time the ball hits the paddle your score increases. If the ball falls off the bottom edge of the screen the game ends (“Game Over”). Your goal is to get as many points as possible before the game ends.

What you will need:
- A computer
- Internet connection

What you will learn:
- Understand the flow of instructions in a program (event > loop > condition > action).
- Use sensors (for example “touching paddle”), variables (for example “score”), and game messages (for example “Game Over”).
- Connect the movement of the ball, the control of the paddle, the bouncing logic and the limits of the game field.
- Apply a simple method of troubleshooting: identify symptom > find cause > apply fix.
- Develop logical thinking, care for sequence of actions and confidence in testing and refining code.

Let’s Code the Game!

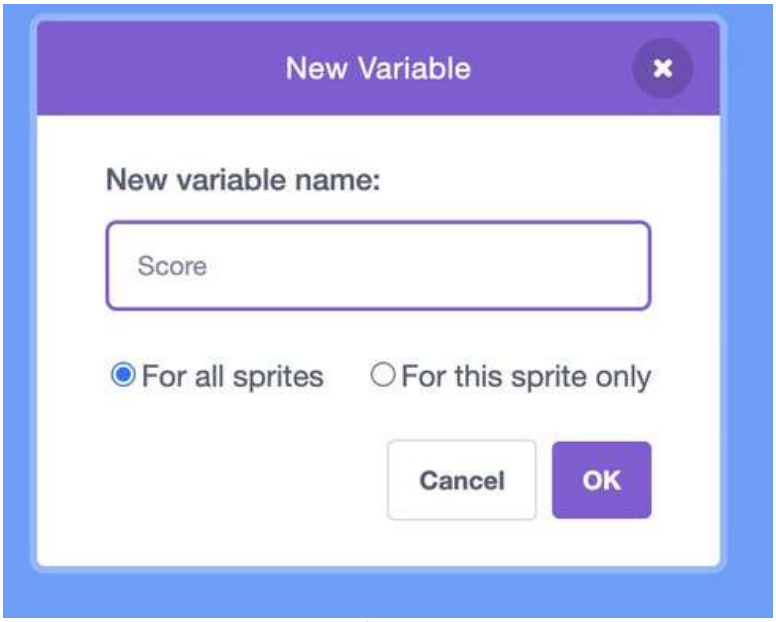
Funded by the European Union



Screenshot 1

Backdrop and Sprites

1. Open Scratch and create a new project.
2. Choose a backdrop for the game field.
3. Add a sprite for the ball, calling it “Ball”.
4. Add a sprite for the paddle, calling it “Paddle”.

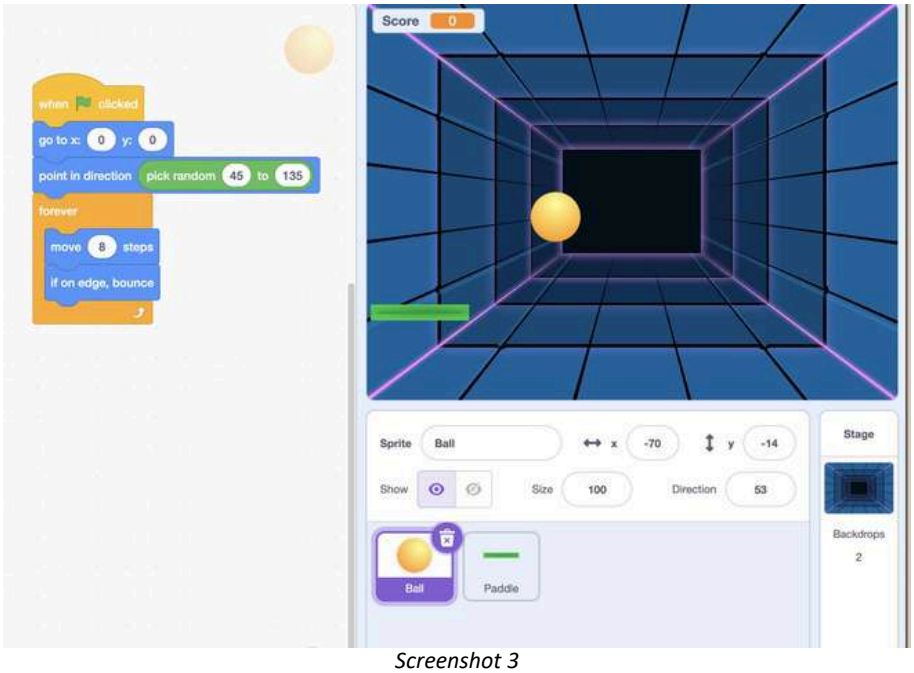


Screenshot 2

Funded by
the European Union

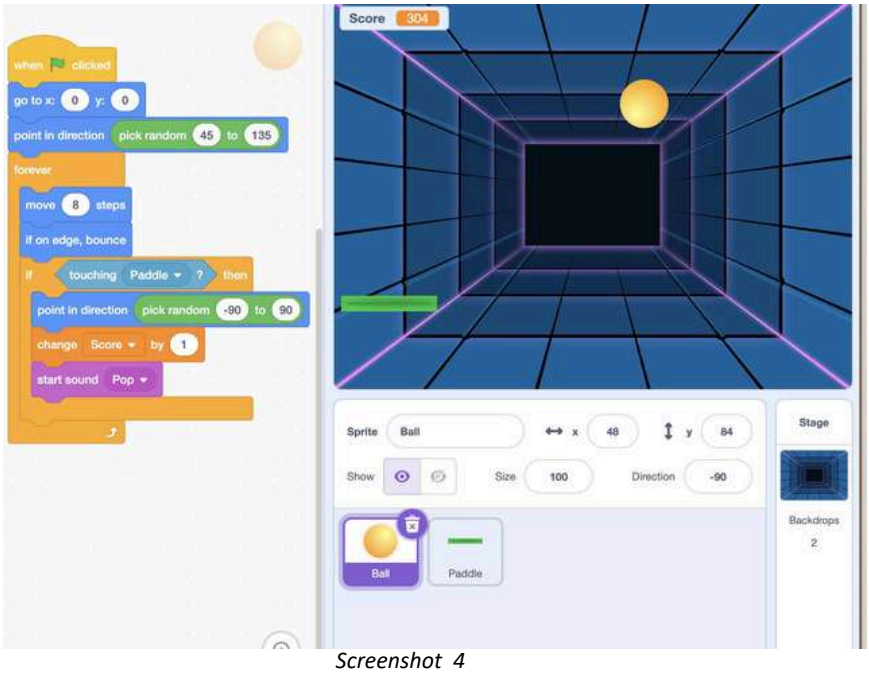
Set the variable
1. Create the variable
- clicking on Variables section
- clicking on “Make a Variable”
- rename the variable calling it “Score”
- select “For all sprites”
This variable will keep track of how many times the ball hits the paddle.
When you run the game, the score will appear at the top-left corner of the screen.

Code for the Paddle
The paddle needs to move left and right to follow the mouse (or keyboard).
Click on the Paddle sprite and add these blocks:
1. From Events, drag “when green flag clicked”.
2. From Motion, drag “go to x:0 and y:-100”, to set the initial positin
2. From Control, drag a forever loop.
3. From Motion, drag “set x to ” and place it inside the forever loop.
4. From Sensing, drag “mouse x” next to “set x to”
Now the paddle will always move horizontally with your mouse when you run the game.



Screenshot 3

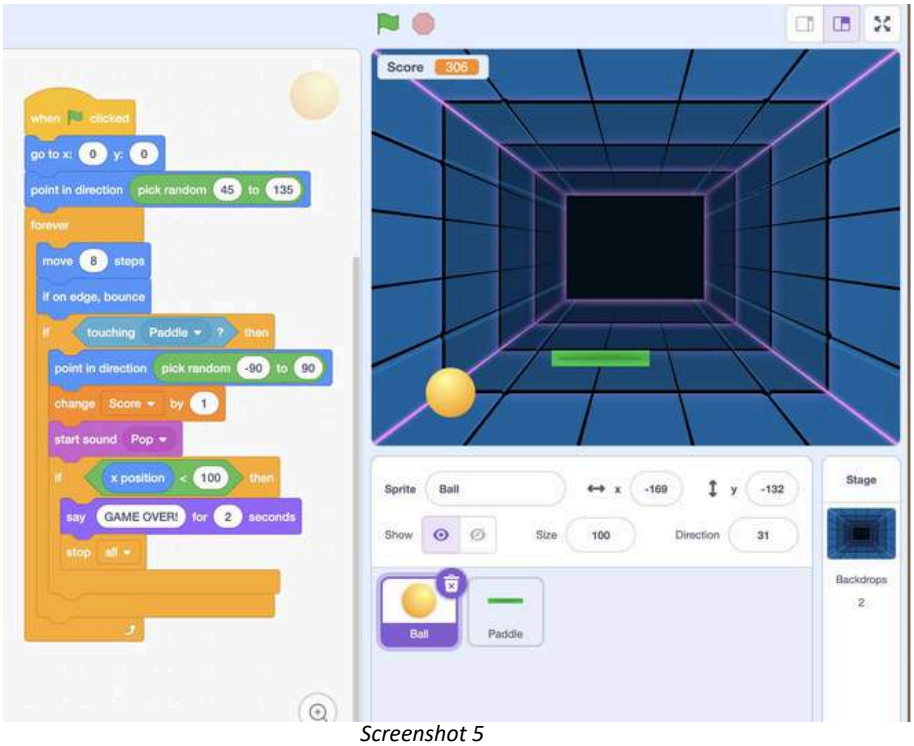
Code for the Ball
Click on the Ball sprite and add these blocks:
1. The motion block “When green flag clicked”
2. “go to x: 0 y: 0” (from Motion): this resets the ball to the center.
3. Point in direction (pick random 45 to 135) (from Motion and Operators): this makes the ball start in a random downward direction.
4. Add the block “forever” (from Control)
Inside the forever loop, add:
- move 8 steps (from Motion)
- if on edge, bounce (from Motion)
Now, when you click the green flag, the ball will move continuously and bounce off the screen edges.



Screenshot 4

Make the ball bounce on the paddle and update the score
Stay in the Ball sprite.
Inside the same forever loop, below “if on edge bounce”, add this new block group:
1. if touching [Paddle]? (from Sensing)
Inside that “if” block:
- point in direction from motion > inside this block, add “Pick random -90 to 90” from operators
- change score by 1 (from Variables)
- play sound [pop] (from Sound)

Funded by the European Union



Screenshot 5

Add "Game Over" when the ball falls off the screen

Still on the Ball sprite, inside the forever loop, add another condition below the previous blocks:

1. Add the if block from Operators

-inside the if block, add "<" from operators

- inside the "<" operator add "x position" from Motion

2. Inside that "if" block:

- say "Game Over" for 2 seconds (from Looks)
- stop all (from Control)

This stops the entire game when the ball drops below the paddle area.

Test your game

1. Click the green flag to start.
2. Move your mouse left and right to control the paddle.
3. Watch the ball move and bounce off the walls and paddle.
4. Check that the score increases when the ball hits the paddle.
5. Let the ball fall off the bottom edge to see if the game ends correctly.

Funded by the European Union

Troubleshooting check-list

When your Pong game does not work correctly, follow these steps one by one. Do not skip steps. Check everything carefully and test again after each action.

Order	What to check	Why
1	Check if the green flag was clicked.	Sometimes the game simply did not start. Click the green flag again.
2	Check which sprite is selected.	Make sure you are adding blocks to the correct sprite (for example, Ball, Paddle, or Score).
3	Check that all scripts start correctly.	Every sprite that does something should begin with "when green flag clicked."
4	Check that loops are running.	Check that loops are running.
5	Check the order of your blocks.	The computer follows your instructions from top to bottom. If something happens too soon or too late, move the blocks to a better position.
6	Check conditions (if blocks).	Make sure each "if touching … then" block uses the correct sprite name (for example "if touching Paddle").
7	Check positions.	If the ball disappears, look at its x and y position. Add a temporary block "say (join y: (y position)) for 1 second" to see where it is.
8	Check variables.	Show your variable (for example "score") on the screen and see if it changes during the game.
9	Check directions and speed.	If the ball moves too fast or too slow, change the number in the "move steps" block or change its starting direction.
10	Restart and test again.	After each fix, click the green flag again and test. Watch what changes, and stop when everything works correctly.
11	If nothing works, ask for help.	Explain clearly what you tried and what you saw. Example: "The ball moves but does not bounce on the paddle. I checked the if touching block and it is there."

Funded by the European Union

6. CONCLUSIONS

You have now completed this module, a journey through clear communication, visual learning, and creative practice. Along the way, you learned how simple words, strong visuals, and step-by-step actions can make even complex ideas easy to understand. You built, coded, and experimented, discovering that learning becomes exciting when you can see and touch your progress.

Keep exploring, keep building, and keep asking “What happens if I try this?”.
Each new step is a chance to learn something amazing. Your journey does not end here, it continues with every light you turn on, every block you connect, and every idea you bring to life.

If you had fun, explore the other modules of the learning course!

Funded by
the European Union

7. FURTHER READING

If you had fun and want to explore more, here are some great places to look online. You can ask your educator to help you find them.

- **Scratch:** https://scratch.mit.edu

"https://scratch.mit.edu" Scratch lets you create games, animations, and stories by using visual blocks. It is the same kind of coding we practised, simple and fun for everyone.

- **Code.org:** https://www.code.org

This site offers short and easy lessons about coding. You can play games, move characters, and learn how computer logic works step by step.

- **Arduino Education:** https://education.arduino.cc

Arduino helps you understand electronics through real-world projects. It is great for learning how sensors, lights, and motors work together.

- **MakeCode by Microsoft:** https://makecode.microbit.org

"https://makecode.microbit.org" This site lets you try block-based coding to control a small device called a Micro:bit. It is a great way to practise what you learned about programming flow.

- **Instructables:** https://www.instructables.com

"https://www.instructables.com" Here you can find thousands of creative do-it-yourself projects, from simple circuits to recycled crafts. Each project includes photos and step-by-step guides.

- **Exploratorium – Tinkering Studio:** https://www.exploratorium.edu/tinkering

A fun and inspiring place to learn by doing. You can find ideas for building, experimenting, and creating projects that mix art, science, and play.

8. SELF-ASSESSMENT GRID

Let's see what you remember! Choose the best answer for each question. The correct answer is in **bold**.

Nr.	Question	A	B	C	D
1	Why is clear communication important in electronics and programming?	Because electronics always work even with confusing instructions	**Because small mistakes or unclear steps can stop things from working**	Because it makes the project look more professional	Because it makes coding faster but harder to understand
2	What is the main goal of using simplified language in this module?	**To make complex ideas easier to understand**	To remove all technical words	To write as little as possible	To make the text sound more scientific
3	Why are visual aids like pictures and diagrams useful?	They make the text colorful and fun	**They help learners remember and understand better**	They replace teachers and explanations	They make the page look full
4	What does "block-based programming" mean?	Writing code using long lines of text	**Building code with colored blocks that fit together**	Drawing shapes to tell the computer what to do	Typing special symbols and numbers
5	What should you do before starting a project?	Begin building immediately	Read all the steps at the end	**Look at and organize your materials on the table**	Ask someone else to prepare everything

Funded by the European Union

Nr.	Question	A	B	C	D
6	What happens in the “Catch the Mouse” activity?	**The cat chases the mouse until it catches it**	The player draws a maze for the mouse	The mouse collects cheese to win	The player builds the game using real wires
7	What is troubleshooting?	**Finding and fixing problems when something doesn’t work**	Writing faster code with fewer steps	Starting a new project every time you make a mistake	Changing colors and shapes in the program
8	What is the first step when troubleshooting a coding problem?	Delete the code and start again	**Drawing pictures**	Colorful blocks	Add more blocks without checking
9	What skill do learners practise when following step-by-step instructions?	**Logical thinking and sequencing**	Reading stories faster	Drawing more creative pictures	Listening to music while coding
10	What is the main message of the module’s conclusion?	Stop experimenting once you finish the lesson	**Keep exploring, building, and learning through curiosity and practice**	Use only one method and never change it	Memorize every code block and definition

Funded by the European Union

MODULE 3: LEARNING TOGETHER: CONNECTING & GETTING SUPPORT

Funded by
the European Union

Overview

This module focuses on the importance of learning as a collective experience rather than an individual one. For adults with Down syndrome and other intellectual disabilities, the process of acquiring new skills—particularly in technical domains such as electronics and coding—can be challenging. However, research and practical experience both demonstrate that collaborative learning environments significantly enhance not only knowledge acquisition but also confidence, independence, and social integration.

The FEAT-DS project, through its comparative research, revealed that adults with Down syndrome often thrive when they can rely on peer-to-peer support, clear guidance from educators, and structured opportunities for teamwork. In such environments, learning becomes a shared journey. Each participant contributes according to their strengths, and the group as a whole benefits from diversity in ability, creativity, and perspective.

The central themes of this module—teamwork, asking for help, and embracing mistakes as part of learning—are rooted in inclusive education practices. The module provides theoretical grounding for educators and practical activities for learners. It highlights the idea that technical skills cannot be developed in isolation; rather, they must be nurtured alongside soft skills such as communication, collaboration, problem-solving, and resilience.

Importantly, the module is designed with accessibility in mind. This includes simplified language, pictogram-based explanations, and visual supports. The inclusion of reflection moments and collaborative games ensures that participants not only acquire technical knowledge but also internalize key social and emotional competencies.

This module invites us to move from individual practice to collaborative teamwork, learning how to create, solve problems and celebrate success together!

Funded by
the European Union

Aims

The module aims to achieve the following:

- Develop learners' teamwork and collaboration skills while working on coding and electronics projects.
- Encourage a positive approach to mistakes, highlighting that errors are natural, expected, and essential for progress.
- Strengthen learners' capacity to seek help constructively from peers, educators, and resources.
- Reinforce the importance of peer support, educator guidance, and group reflection in creating a safe, inclusive learning environment.
- Provide educators with concrete tools and strategies to facilitate collaborative and supportive learning.
- Equip learners with a sense of self-efficacy and confidence, encouraging them to continue exploring digital and technical fields independently.

Expected Outcomes

By the end of this module, learners will be able to:

1. Identify the value of teamwork in learning contexts and describe how working together improves problem-solving.
2. Demonstrate basic help-seeking behaviors, such as raising a hand, using pictograms, or directly asking peers/educators for clarification.
3. Recognize mistakes as part of the learning process, reducing anxiety and fostering resilience.
4. Participate actively in collaborative projects, fulfilling different roles within a group.
5. Troubleshoot basic coding and electronics problems in partnership with others.
6. Engage in structured reflection about their experiences, identifying challenges, successes, and moments where help was beneficial.
7. Show improved confidence and motivation in pursuing further technical and digital learning opportunities.

Funded by the European Union

1. THE VALUE OF TEAMWORK IN LEARNING

Teamwork is widely recognized as a cornerstone of inclusive and effective education. For learners with Down syndrome, the benefits of teamwork extend beyond academic performance to include social, emotional, and personal development.

Research on collaborative learning emphasizes that when learners engage in tasks together, they:

- Share cognitive load: Difficult tasks become more manageable when responsibilities are distributed.
- Learn from diverse perspectives: Different learners notice different details, leading to more creative solutions.
- Develop communication skills: Explaining an idea to others requires clarity, patience, and practice.
- Build social connections: Working together fosters friendships and combats isolation.

In the context of coding and electronics, teamwork allows participants to:

- Assign different roles (e.g., connecting circuits, reading instructions, entering code).
- Provide immediate peer feedback.
- Celebrate success collectively, reinforcing motivation.

Educators should emphasize that there is no “best” role in a team. Each contribution is valuable. This message reduces competitive pressure and reinforces inclusivity.

Funded by the European Union

Practical Activity 1 — Pair Build: Light-Touch Sensor

Purpose
Learners practice teaming up, sharing roles, and safely building a tiny circuit that gives instant feedback (light/sound). This uses the project's tinkering + coding approach and Easy-to-Read supports.

Duration
~90–120 minutes (pair work + group share-out).

Materials (per pair)
- Battery pack (2xAA or 3V coin cell holder)
- On/off switch or push button
- 1 LED (with resistor or pre-wired) or a small buzzer
- 6–8 colour-coded jumper leads
- Optional: simple foil "touch pad" (two pieces of aluminium foil + wire)
- Printed step cards with large photos/pictograms (A5)
- Role cards: Builder / Helper
- Large icons/pictograms

Safety rules (teach & post)

1. **Power OFF before changing wires.**
2. Keep leads tidy; no loose metal touching the battery.
3. Ask before tools. Then switch ON to test.
   (Aligns with project guidance on concrete, step-by-step learning and visual cues.)

Accessible setup
- One idea per step; big photos; minimal text; icons for Power, Wire, Test.
- Short demo first so learners see cause → effect (LED/buzzer responds instantly).

Mini-lesson (10–12 min)
1. Show a simple battery → switch → LED circuit. Turn ON: light. Turn OFF: no light.
2. Introduce roles: Builder (hands on parts), Helper (reads card, checks steps). Switch roles halfway.
3. Normalize mistakes: "If it doesn't work, we check: power → wires → part." (3-step debug used across modules.)

Objective (for learners)

- Work in a pair using roles; build a tiny circuit; test and explain what it does.
- Ask for help using a simple help-script; try the 3-step debug.

Time & grouping

- 10 min demo & setup → 25–35 min build & test → 10–15 min share → 10 min clean-up/reflection.

Materials

As listed above; optional foil touch pad to replace button (pressing two foils with a finger closes the circuit).

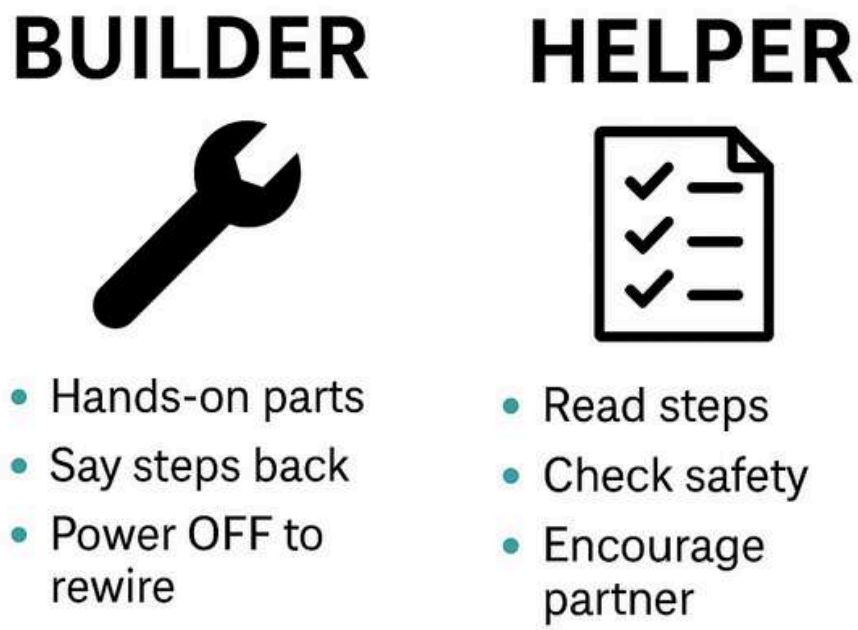


***Figure 1a**. Role cards (Builder / Helper)*
Cards for pair roles and turn-taking.

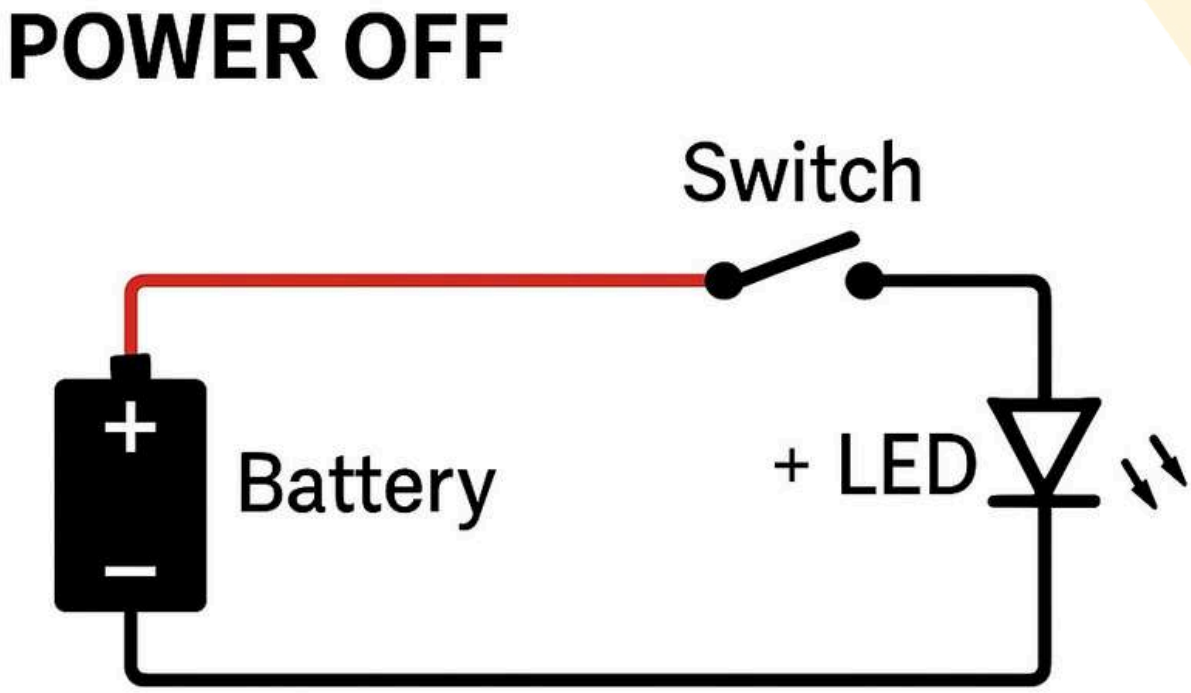


***Figure 1b**. Battery → Switch → LED wiring (color-coded)*
Cards for pair roles and turn-taking.

Step-by-step (print on A5 cards with photos)

1. **Choose roles.** Builder wears the "Builder" card; Helper reads the step card. (Switch later.)
2. **Power OFF.** Connect battery holder to switch (red to switch in; black to LED/buzzer "–").
3. Connect switch to LED "+" (or buzzer "+").
4. **Check wires: red → switch → LED (+), black → LED (–) →battery (–).**
5. **Power ON → Test.** Do you see light / hear sound?
6. **Switch roles** and rebuild once.

(Keep wording short, one idea per line; match icons to verbs.)

Funded by the European Union

Troubleshooting card (3-step debug)
A) **Power**: Is the battery connected? Switch ON?
B) **Connections**: Any loose or crossed leads? Follow colour order.
C) **Component**: Try another LED/buzzer.
(Instant sensory feedback confirms success.)

Help-seeking micro-script (card)
"I tried ___. The problem is ___. Can you look at ___?"
(Use of simplified language and localized supports.)

Differentiation (for mixed abilities)

- **Easier**: Use pre-wired LED module; give a one-page photo-only guide.
- **Harder**: Replace button with foil touch pad; explain why touching joins the circuit.
- **AAC supports:** pictograms for actions (connect / press / test); gesture cues.

Facilitator notes (practical)

- Model **Builder/Helper** talk: Helper reads; Builder repeats and does. Swap mid-way.
- Praise **process** (checking steps) not just result; celebrate first successful light/sound.
- Keep a spare battery/LED kit at hand to isolate component faults quickly.

Mini-check (during activity)
Ask a pair to point to power, switch, load (LED/buzzer) and say in one line what happens when the switch is ON. (Concrete identification supports understanding.)

Role rotation

- Mid-session switch: Helper becomes Builder; Builder becomes Helper.
- Each pair ticks a **"We switched roles"** box on the step card (visual accountability).

Peer support
• When stuck for 2 minutes: **try one debug step**, then use the **help-script** with a neighbour or educator. (Builds resilience and structured support.)

Funded by the European Union

Share-out (Gallery, 10–15 min)

- Pairs place circuits on a table. Another pair presses button/touch pad to see/hear it work.
- Use **Two stars & a wish**: say two positives and one idea to try. Keep it short and kind. (Fosters constructive feedback.)

Quick reflection (exit ticket)

Learners complete two icons on a half-sheet:
"What worked for us" (draw or 3–4 words)
"What to try next time" (draw or 3–4 words)

Observation checklist (facilitator, tick per pair)

□ Took roles and switched once
□ Followed **power-OFF** rule when changing wires
□ Built a working circuit (light/sound)
□ Used 3-step debug before asking for help
□ Used the help-script at least once

Cleanup & storage (teach routine)

- Power OFF; remove battery; untangle and bundle leads with a twist tie; return parts to a labelled box.
- Take a photo of the final setup; note any broken parts.

Figure 2. *Example "Two stars & a wish" feedback tokens (three pictograms).*

Funded by the European Union

2. ASKING FOR HELP: A STRENGTH, NOT A WEAKNESS

One of the most important shifts in mindset is viewing mistakes not as failures but as opportunities. This concept is sometimes called a "growth mindset" in educational theory. For learners with Down syndrome, who may already feel less confident in academic or technical domains, reframing mistakes is especially empowering.

In electronics and coding, mistakes are inevitable:

- A wire may be connected incorrectly.
- A line of code may be missing.
- A sensor may not respond as expected.

Instead of discouragement, these moments can become powerful teaching opportunities. Educators should emphasize:

- Mistakes are normal.
- Everyone makes mistakes—even experts.
- Mistakes help us see what we need to try differently.
- Fixing mistakes is how we get stronger.

Example (educator speaking to group):
"When the light doesn't turn on, it doesn't mean we failed. It means we learned something new about the circuit. Now we can check the connections together."

This perspective shifts the classroom culture from fear of failure to excitement about discovery. Furthermore a recurring challenge in inclusive education is that many learners with intellectual disabilities hesitate to ask for help. This may stem from past experiences of feeling stigmatized, or from a desire to appear independent. However, help-seeking is not a weakness—it is an essential skill for lifelong learning.

Educators can normalize help-seeking by:

- Explicitly teaching phrases or gestures that learners can use.
- Reinforcing positively when learners request clarification.
- Modeling help-seeking behaviors themselves (e.g., an educator openly checking a manual and saying, “I need help too sometimes”).

Help can take many forms:

- Asking a peer for assistance.
- Requesting a demonstration from the educator.
- Using a pictogram or visual card to signal confusion.
- Pausing and asking the group, “Can someone explain this step again?”

The key is that learners understand asking for help is not giving up—it is part of succeeding.

Practical Activity 2 — Troubleshooting Together

First part:

Purpose
Learners practice how to ask for help and how to fix problems in a small electronics-plus-code task. We teach a clear, repeatable debug routine and simple language support. Immediate light/sound feedback shows success.

Duration
~90–120 minutes (demo → paired co-debug → gallery share).

Materials (per pair)

- Laptop/tablet with a block-based coding tool (beginner-friendly, large icons)
- USB microcontroller or simulator
- LED and/or buzzer circuit from Practical Activity 1
- Colour-coded jumper leads
- Printed debug card & help-script (A5, pictograms)
- Optional: projector for live code demo. (Use big icons, step-by-step visuals, localized Easy-to-Read text.)

Safety baseline (repeat)
Power OFF before moving wires; keep leads tidy; ask for tools; test only after checks. (Concrete, step-by-step habits.)

Mini-lesson (10–12 min)

- Show a short block program that should blink an LED or beep a buzzer.
- Name today's two skills: Ask for help clearly + Use 3-step debug.
- Model both with a broken example (e.g., wrong pin or block order). Tie to project's tinkering + coding method.

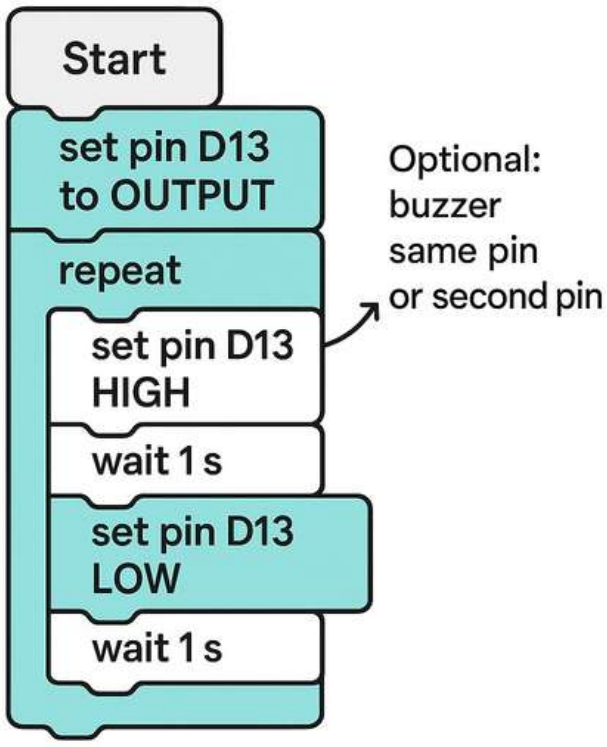


Figure 3. *Example "Blink & Beep" blocks (Start → Set pin → Repeat: On/Wait/Off/Wait).*

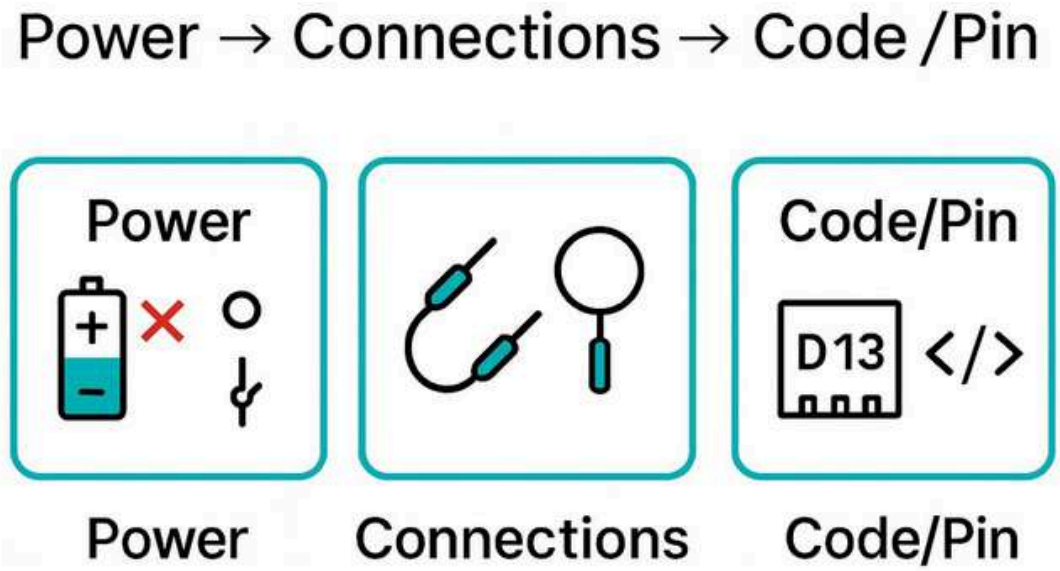


Figure 4. *pictograms: Power → Wires → Part/Pin (check in this order).*

```
void setup() {
  pinMode(13, OUTPUT);
}
void loop() {
  digitalWrite(13, LOW);
  delay(1000);
   // Optional: buzzer
   same pin or second pin
}
```

Figure 5. *Debug card pictograms.*

Second part:

Objectives (for learners)

- Ask for help using a simple script.
- Apply the 3-step debug to make code + circuit work.
- Say what each main block does in one short sentence.

Time & grouping

- 10–12 min demo → 30–40 min paired co-debug → 10–15 min gallery share → 10 min reflection.

Setup

Give each pair a **buggy starter project** (two typical bugs below). Provide the **Practical Activity 1 circuit** ready to connect. Interfaces should use **large icons** and clear layout.

Starter bugs (choose 1–2)

A) **Wrong pin** (LED on pin D13 but code sets D12)
B) **Missing "set pin mode/output"** block (or wrong block order)
C) **Too short wait** (human can't see blink)
(Use bugs that create **visible/audible** effects when fixed.)

Step-by-step (print on A5 with photos/icons)

1. **Look & tell**: What should the program do? (blink/beep)
2. **Connect** the circuit (as in Practical Activity 1). **Power OFF** while wiring.
3. **Run** the code. What happens? (say it)
4. **3-step debug**: **Power → Connections → Code/Pin**.
5. **Change one thing**, test again.
6. When it works: **say what you fixed**; take a photo.
   (Keep text short, one idea per line; pair it with pictures/pictograms.)

Help-seeking micro-script (cards)

"I tried ___. The problem is ___. Can you look at ___?"
(Use simplified, localized language + symbols.)

Facilitator moves (practical)

• Cue the **help-script** before stepping in; point to the 3-step card.
• When pairs stall, ask: "What did you **change**? What will you **try next**?"
• Celebrate **process** (testing one change at a time) and **sensory success** (light/sound).

Funded by the European Union

Differentiation
- **Easier:** Provide a pin-label photo and highlight the correct block stack.
- **Harder:** Add a second output (LED + buzzer) or a touch pad trigger.
- **AAC supports:** Pictograms for actions; gesture prompts for “test/change.”

Mini-check (during activity)
Ask a learner to point to the pin used in code and on the board; say “This block turns the LED on.” (Short verbal + visual pairing.)

Team routines for help & fix
Framing a good question
- Show the broken behaviour → say what you expected → show the line/block you think is the problem.
- Try one change before asking again. (Builds resilience and structure.)

Peer-assist protocol (2–3 min)
Neighbor listens → repeats the problem → suggests one debug step → watches the test. Keep voices low; use the help-script if stuck. (Creates a micro “community of practice”.)

Share-out (Gallery, 10–15 min)
- Each pair runs the fixed program; peers press the button/touch pad to see/hear it respond.
- Feedback: **Two stars & a wish** (two positives + one idea). (Positive, concise language.)

Reflection
“What we fixed” (draw/3–4 words)
“Next time we will try…” (draw/3–4 words)

Observation checklist (facilitator)
□ Used the **help-script** once
□ Followed **3-step debug**
□ Identified the **correct pin/block**
□ Verified fix with light/sound

Funded by the European Union

3. THE ROLE OF EDUCATORS AND PEERS

Educators in this module serve as facilitators rather than just instructors. Their tasks include:

- Creating a safe, inclusive atmosphere.
- Providing clear, step-by-step guidance.
- Modeling collaboration and help-seeking.
- Using accessible materials (pictograms, simplified texts, diagrams).
- Encouraging reflection after activities.

The educator is not the sole “expert” but a co-learner, showing that even adults continue to learn and adapt.

Peer interaction is equally crucial. In fact, peers often provide explanations in simpler, more relatable language. Peers may also notice errors educators miss. Importantly, peers offer emotional support—a smile, encouragement, or applause can boost confidence more than technical guidance.

Peers should be encouraged to:

- Support each other verbally (e.g., “Good job!” or “Let’s try again”).
- Share tasks equitably.
- Celebrate group achievements.

This dynamic transforms the group into a community of learners, where everyone contributes and everyone benefits.

Practical Activity 3 — Show and Share Gallery

Objectives (for learners)

- Demonstrate a working circuit/program and state in one sentence what it does.
- Use a simple feedback script with a partner pair (two stars & a wish).

Time & grouping

- 10–15 min set-up → 30–40 min gallery walk (pairs rotate) → 10–15 min feedback circle → 10 min reflection.

Set-up

- Each pair places their circuit on a display mat.
- They prepare a **caption strip**: "Our Project: _____. It does: _____." (Easy-to-Read).

Feedback scripts (cards)

- **Two stars & a wish:** " I like _____; ; Next time _____."
- **I notice / I wonder:** "I notice _____. I wonder _____."
  (Use if learners prefer observation language.)
  (Keep phrasing short; pair with icons.)

Step-by-step (A5 card for learners)

1. **Try it**. Press button/touch pad → watch **light/sound.**
2. **Say what happened** in one line.
3. **Place tokens:** two + one .
4. **Hosts** say thanks and **log one idea** on their mat. (Immediate sensory feedback keeps motivation high.)

Differentiation

- Easier: Provide sentence starters and picture-only tokens.
- Harder: Ask pairs to compare two versions (before/after fix) and explain the change.
- AAC: Symbol cards for "like/idea/works."

Facilitator actions

- Prompt concise talk: "One sentence." "Show with finger where it changes."
- Re-ground in criteria: "Does the sensor react as planned?" "Can you name the fix?"

Figures (placeholders)

Funded by the European Union

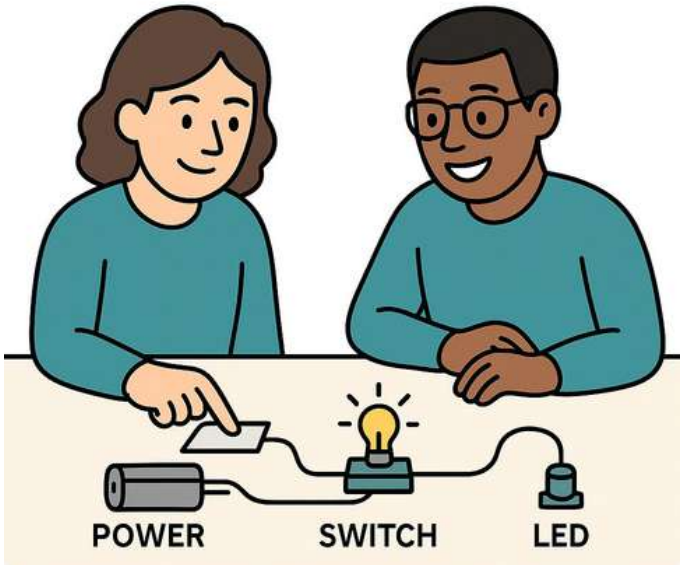


Figure 7. *Photo of peers testing a touch sensor; light turns on.*

Reflection, celebration & documentation
Reflection circle (8–10 min)
Go round quickly:
"One thing that worked for us was _____."
"Next time we will try _____."
(Short turns + pictograms help fluency.)

Mini-check (during/after gallery)
Ask any learner to:
- Point to **power/switch/output** and say what happens when ON.
- Name **one change** they made to fix the program/wiring. (Concrete identification supports understanding.)

Observation checklist (facilitator)
□ Demonstrated a **working** circuit/program (sensor reacted)
□ Gave **two positives** and **one idea** to peers
□ Stated what the project **does** in one sentence
□ Logged **one improvement idea** on mat
(Use for step-by-step assessment during piloting & reporting.)

Celebration
- Applause wave + group photo; optional printed "We shared our work!" badge.
- Keep language **simple & affirmative**; reinforce autonomy and success.

Funded by the European Union

Evaluation & Self-assessment

Nr.	Question	A	B	C	D
1	When changing wires you should first…	Switch tools	**Turn power OFF**	Ask a friend	Press the button harder
2	What is the first step of our debug routine?	**Check power**	Edit the code	Change the LED	Replace all wires
3	The Helper role mainly…	Uses tools only	**Reads steps, checks safety, supports**	Sits and waits	Takes photos only
4	Your LED is on D13, but code sets D12. Best fix?	Swap the battery	**Change code to D13**	Change LED color	Clean the table
5	Which block makes an LED turn on?	Wait 1 s	Repeat forever	**Set pin to ON/High**	Start/When green flag clicked
6	Before asking the educator, you should…	**Try one debug step**	Start again	Change two things	Pack up
7	Safe clean-up includes…	Storing with battery connected	**Power OFF, remove battery, tidy leads**	Mixing parts in one box	Leaving wires loose
8	“Two stars & a wish” means…	Two problems + one fix	Only ideas	Only praise	**Two positives + one idea**
9	A good help-ask includes…	“It’s broken.”	“Please fix it.”	**What you tried, the problem, where to loo**	Silence and pointing
10	Our success criterion in this module is met when…	The code looks long	We finish first	**The sensor/circuit behaves as intended and we can explain it**	The table is neat

Funded by the European Union

Further Reading

Accessibility & Easy-to-Read / Leichte Sprache
• Inclusion Europe — Information for all: European standards for making information easy to read and understand.
https://www.inclusion-europe.eu/easy-to-read-standards-guidelines
inclusion-europe.eu
• Netzwerk Leichte Sprache — Die Regeln für Leichte Sprache (Neuauflage 2022, PDF).
https://www.netzwerk-leichte-sprache.de/fileadmin/content/documents/regeln/Regelwerk_NLS_Neuauflage-2022.pdf
netzwerk-leichte-sprache.de
"https://www.netzwerk-leichte-

IFLA — Guidelines for Easy-to-Read Materials (2nd ed., PDF)
https://www.ifla.org/files/assets/hq/publications/professional-report/120.pdf
ifla.org

Pictograms & AAC (for cards, mats, scripts)
• ARASAAC symbol set — searchable, free pictograms with online editor.
https://beta.arasaac.org/pictograms/search?tab=1
beta.arasaac.org

Electronics & block-based coding (beginner-friendly)
• Arduino — Blink (official tutorial; quickest hardware feedback).
https://www.arduino.cc/en/Tutorial/Blink
Arduino

• Tinkercad Circuits — Lesson 1: Blinking LED (no hardware needed; classroom-safe simulator).
https://www.tinkercad.com/things/bDHquWl5k2q-lesson-1-blinking-led
Tinkercad

• micro:bit — Get coding with MakeCode (block-based editor + device pairing).
https://microbit.org/get-started/getting-started/get-coding
microbit.org

• Microsoft MakeCode — micro:bit Getting Started (browser editor; big blocks).
https://makecode.microbit.org/tutorials/getting-started
Microsoft MakeCode

• Adafruit Circuit Playground Express — Quickstart (rich light/sound board; great for sensory feedback).
https://learn.adafruit.com/circuit-playground-express-circuitpython-5-minute-guide/overview
Adafruit Learning System

Inclusive design for teaching (planning lens)
• CAST — UDL Guidelines 3.0 (optimize choice, multiple representations & actions).
https://udlguidelines.cast.org
udlguidelines.cast.org

• UDL 3.0 graphic organizer (accessible PDF).
https://udlguidelines.cast.org/static/udlg3-graphicorganizer-digital-nonumbers-a11y.pdf
udlguidelines.cast.org

Funded by the European Union

References

Adafruit. (2019). *Circuit Playground Express: CircuitPython 5-minute quickstart guide.* https://learn.adafruit.com/circuit-playground-express-circuitpython-5-minute-guide/overview Adafruit Learning System

Arduino. (2024, February 10). *Blink.* https://www.arduino.cc/en/Tutorial/Blink Arduino

CAST. (2024). *The UDL Guidelines 3.0.* https://udlguidelines.cast.org

CAST. (2024). *The Universal Design for Learning Guidelines (Graphic Organizer, digital, no numbers) [PDF].* https://udlguidelines.cast.org/static/udlg3-graphicorganizer-digital-nonumbers-a11y.pdf

Inclusion Europe. (2021, October 6). *Information for all: European standards for making information easy to read and understand.* https://www.inclusion-europe.eu/easy-to-read-standards-guidelines / inclusion-europe.eu

International Federation of Library Associations and Institutions. (2010). *Guidelines for easy-to-read materials* [PDF]. https://www.ifla.org/files/assets/hq/publications/professional-report/120.pdf / ifla.org

Microsoft. (n.d.). *Getting started — MakeCode for micro:bit.* https://makecode.microbit.org/tutorials/getting-started (Original work published 2016; content updated frequently.) Microsoft MakeCode for micro:bit

Micro:bit Educational Foundation. (n.d.). *Get coding with MakeCode.* https://microbit.org/get-started/getting-started/get-coding / microbit.org

Netzwerk Leichte Sprache. (2022). *Die Regeln für Leichte Sprache* [PDF]. https://www.netzwerk-leichte-sprache.de/fileadmin/content/documents/regeln/Regelwerk_NLS_Neuauflage-2022.pdf / netzwerk-leichte-sprache.de

Tinkercad. (n.d.). *Lesson 1 — Blinking LED.* https://www.tinkercad.com/things/bDHquWl5k2q-lesson-1-blinking-led / Tinkercad

Zaragoza Government — ARASAAC. (n.d.). *ARASAAC pictograms search engine.* https://beta.arasaac.org/pictograms/search?tab=1 / beta.arasaac.org

Funded by the European Union

MODULE 4: ELECTRONICS IN OUR WORLD: FROM HOBBIES TO POSSIBILITIES

Funded by
the European Union

Overview

This module highlights another important topic, which is called “Electronics in Our World—From Hobbies to Possibilities” and responds to “Fragmentation of Services and Lack of Formal Policy Coordination”. The FEAT-DS project aims to help individuals with Down syndrome acknowledge practical skills in electronics, which will not only improve their self-esteem but also bring them more employment perspectives and help them blend into society.
Firstly, the use of electronics in daily objects will be discussed. Later, the topic will be followed by hobbies and activities related to electronics and coding. As well as this, participants will be asked to brainstorm some ways that could be useful in daily tasks and potentially future roles.
Tinkering and Coding will be the main activities of the practical part. The Tinkering activity aims to construct an electronic device suitable for practical tasks like plant watering reminders. The goal of the activity is to remember that their ideas have value. In Coding, the main aim is to build an electronic project that aligns with users’ input, which could demonstrate the practical relevance of their skills.

Aims

- Learn and raise awareness of how electronics are used in everyday objects.
- Discover the hobbies and activities that are connected to electronics and coding.
- Understand the simple ways electronics skills could be useful in daily tasks.
- Learn how to design and build a simple electronic device by tinkering activity.
- Discover how coding can create interactive electronic projects.

Funded by
the European Union

Expected Outcomes

- You will be able to explore how electronics are used in everyday objects such as phones or remote controls.
- You will be able to learn about hobbies and activities related to electronics and coding.
- You will be able to understand how simple electronics skills could be useful in everyday life.
- You will be able to build a simple device that solves a small, real-world problem.
- You will be able to create an electronic project that responds to a user's input.

Funded by
the European Union

1. ELECTRONICS EVERYWHERE – EXPLORING EVERYDAY OBJECTS

Electronics in Daily Life

Have you ever noticed how you can independently turn on the TV or the lights in the room? Or have you ever thought about how you learned how to charge your phone? Most importantly, did you know that every time you do all these activities, you actually use electronics?!

In this module, we will explore more daily electronics tasks and learn how to build the basics ourselves. Reading is always enjoyable, however in this Module course, it will not be limited to just reading; you will also create your own simple device that can help solve small real-world problems.

Where Can Electronics Be Found?

Back some decades ago, we could not confidently say electronics are everywhere, but nowadays they are part of our daily lives—everywhere. A simple example: when you want to watch a movie or listen to music, you use electronics.

Take a second and think about where electronics can be found. Let's have a look at the examples:

- A lamp that you turn on when you enter your apartment
- A remote you use to adjust AC when you feel too hot or cold
- An alarm that helps to be on time
- A toy car that you can race with your friends by controlling with a remote

All objects mentioned above are activated by using electronics and circuits.

Funded by the European Union

2. EVERYDAY ELECTRONICS – EXPLORE THE ELECTRONICS WE DAILY USED

In our activities, we will firstly explore electronics we use daily and later we will create basic devices for daily use.

The Doorbell

Figure 1. The doorbell switch

This is DOORBELL.

Can you imagine how difficult it would be to try to call someone's name until they open the door? Doorbells make it easier to let someone know we wait behind the door.

How does it work?

- When you press the button (input), it triggers the circuit.
- A speaker makes the sound which we call "output".
- The power is usually provided by a battery or adapter.

Next time you press the button, you will understand how it works.

Funded by the European Union

2. EVERYDAY ELECTRONICS – EXPLORE THE ELECTRONICS WE DAILY USED

In our activities, we will firstly explore electronics we use daily and later we will create basic devices for daily use.

The Remote Control

Figure 2. The remote control

Have you ever dreamed of having superpowers and controlling objects? Remote control does not fully give this power, but it does send invisible signals to another device.

How does it work?

- When you press the tiny buttons, the circuit is triggered.
- A tiny chip inside turns the signal into light from a LED (light-emitting diode) and switches the device on.
- The power is provided by batteries inside.

The Alarm Clock

Figure 3. The alarm clock

This is an ALARM CLOCK.
Could you believe this tiny alarm clock includes lots of tiny electronics, which help it to buzz and wake us up in the mornings? You may ask, do people still use alarm clocks, as many of them set their alarm on their phone? There might not be many people who use it; however, it is still fun to learn how it works!

What electronics does it use?

- A small computer chip to keep time.
- A display screen where the numbers are shown
- A buzzer that creates the sound at the set time

If you want to make sure you are on time, please remember to set your alarm clock!

Funded by the European Union

3. ELECTRONICS AS A PART OF HOBBIES

We've discussed using electronics in our everyday lives, but what if they could also be part of our hobbies? Yes, we can create fun activities using electronics and coding! Let's explore what fun activities we may create by using electronics:

Light-up drawing activity: It is time to use your imagination and create colorful paintings and add a light that can be controlled by an on/off button. Imagine using it with a shiny sun to make it brighter.

Basic Night Light: Have you ever felt like your room gets a little dark at night and you sometimes need a little light before falling asleep? You can integrate a lamp into your bed and attach the button nearby to turn it on whenever you want or add sensors that will automatically turn on when it gets too dark.

Spinning toy: Another fun activity to try! Choose your colors and paper pattern, then add a small battery-operated motor to make it spin. Make sure the paper is round for better rotation and choose your favorite colors. Finally, watch as electricity transforms your paper into something fun!

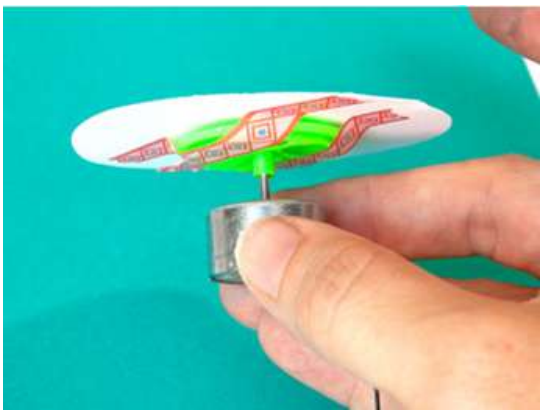


Figure 4. The DIY spinner

[1] Image source: Frog Prince Paperie – "DIY Spinner Art STEM Project."

What role does coding play in our activities? **Coding** simply is a program that tells a device what to do, without using words.

For example: you can code a simple "Hello" message on a screen or LED or write someone's name on it and give it to them. This will not only be a fun activity, but also a creative gift.

Figure 5. The led light board

Now it is your turn. Think, what other fun activities/hobbies could be created by using electronics and coding? Use your imagination and write down your ideas!

Funded by
the European Union

4. PRACTICAL TINKERING ACTIVITIES

Now it is time for the best part: building our own circuits! Remember, this is tinkering. Have fun, explore and see what you can create.

Activity 1: Light Reminder!

In this activity, we will make a small light that will remind you to perform certain daily tasks.

Goal for this activity:

- Build a small lamp that can be turned on and off with a button
- Have a basic tool to help you remember simple everyday tasks, like brushing your teeth

What you will need:

- 1 battery holder with batteries
- 1 LED light (any color)
- 1 on/off switch
- 3 wire cables
- 1 piece of cardboard for mounting
- 2 wire cables Cardboard for mounting

Figure 6. Battery holder with batteries

Figure 7. LED lights

Funded by the European Union

Figure 8. ON/OFF switch

Figure 9. The wire cables

Figure 10. The cardboard

Let's make it together:

1. Take your batteries and place them in the holder
Pay attention to red and black wire of battery holder

2. LED usually has two legs: long leg which is positive (+) and short leg (negative one)

3.Wire 1: Battery (+) on/off switch
There are two ends of each wire end. Firstly, start with the one end of the wire and clip it to the + wire (red one) from the battery holder. And attach the second end to one metal leg which is located behind the switch.

4.Wire 2: Switch LED (+)
In this step, we will connect the long leg (+) from LED with a second wire from the other leg of switch.

5.Wire 3: LED (-) Battery (-)
Take your third wire and connect it from the LED's short leg to the battery holder's black (-) wire.

6. Let's make sure it is safe!
Let's make sure everything is safe before we switch it on. Ask for your supervisor or responsible people to check whether parts are correct and nothing is attached to the wrong part.

7. Turn it on and later off with the switch button!

Funded by
the European Union

Success!

Now let's move on to our last item: the cardboard. You can decorate it with different stickers or draw a picture of what it represents! Imagine a small reminder on your refrigerator, but in an electronic version. As long as you don't turn off the light, the task isn't done. Every time you see the light on, you'll remember your daily actions, like watering the plants or brushing your teeth!

What did we learn?
We've learned that a circuit can help us in our daily lives. When the components are connected correctly, the LED lights up and can serve as a reminder for daily tasks.

Challenge Yourself!

- Can you guess what color is better for reminders and try using different color LED?
- What other ways do you think we could create for reminders?

Activity 2: Your Personal Name Tag

Sometimes in our daily lives, we feel awkward approaching people and introducing ourselves. What if we used a smart badge that told us who we are? In this activity, we'll create a personal badge that lights up at the touch of a button.

Goal for this activity:

- To use a switch to light the tag on.
- To create a tag with light to make it visible

Funded by the European Union

What you will need:
Most of the parts we remember from the first activity, let's remember them again:

- 1 battery holder with batteries
- 1 LED light (Choose your favorite color)
- 1 small switch or push button
- 3 wires with clips (preferably different colors for clear instructions)
- 1 piece of cardboard (adjust the size for name tags)
- Markers or stickers for decorating the cardboard
- glue (to attach the device to the cardboard)

Figure 11.Battery Holder with Batteries

Figure 12. Cardboard

Figure 13. ON/OFF switch

Figure14. 3 Wires with clips

15. Glue

Figure 16. Markers and stickers for decoration

Funded by
the European Union

Time to build it step by step:

1.Let's start with the name tag!
Take the cardboard and cut out a small piece (your name should fit inside) and write on it with a marker. To add color, use stickers and decorate as you wish.

2. Decide what letter or part the light will go
Time to choose what part of your name tag will light up and make a small hole where LED legs will go through.

3.Place the LED
Remember what leg of the LED is positive or negative first, then push LED through the hole and adjust the legs so that they can stay stable.

4.Connect all three wires

- Attach the red wire first from the battery holder to a leg behind the switch.
- Attach another wire to the long leg of the LED (positive one) which will go through the other side of the switch.
- Black wire from the battery holder should be attached to the third wire from the short leg (negative one) of LED

5. Make sure it is safe and turn it on!
Remember to ask your supervisor to check that all parts are secure and securely attached. Also ask for help gluing or taping the parts to the label so they don't fall off.

Success!

Congratulations! You now have a light-up badge. You can wear it at events or anywhere you want to take the first step to break the ice and introduce yourself!

What did we learn?
We learned that we could use electronics and make fun activities!

Challenge Yourself!

- Can you add another LED and make two letters to light up?
- Can you write another person's name on a tag and make it light up?

Funded by the European Union

5. PRACTICAL CODING ACTIVITY

- Computer with MakeCode (or similar ones)

As in previous Modules, your educator will help you get this set up.

Step by step guide:

1. Open your coding program and choose a new project.
2. In order to tell the light when to start – add a block “when button A pressed”.
3. Add another block “Turn LED on”.
4. Add the time that you would like to keep the light on.

Next, set the “Turn LED off” function.

Ready! Press the button and test your light.

Educator's Note: ask your students where it could be useful- such as a reminder to take some break and rest their hands after writing.

Success!

Well done! You coded a light reminder to help you with basic tasks such as “send a message to your parents” or “take the trash out”.

What did we learn?
We learned that coding can be used in our daily activities, and you can use coding to solve small daily challenges.

Challenge Yourself!

- Can you change the waiting time for 10-15 seconds?

Funded by the European Union

6. SMART IDEAS EVERYWHERE

Now it is your turn to use your imagination! Let's brainstorm and share ideas to understand how electronics and coding are connected to everyday life.

Goal for this activity:

- To revise and find out more ideas on how coding and electronics are connected to our daily lives.
- To share creative and fun activities about future projects.

What will you need:

- A small board or paper to write down ideas.
- Markers or pencils
- You can use some pictures from journals and glue them on paper as well.

Where to start:

1. Think about electronics you use in your daily life. It might be at home or school—anywhere.
2. Write or even use stickers or images of each electronics.
3. Next to each of them, add their names and functions.
4. Think about a new idea for everyday simple electronics helper and add it to "to do list" for future to build

Educator's Note: ask your students to talk about what makes electronics and coding interesting for them.

Funded by the European Union

7. CONCLUSION

Good job! Module 4: Electronics in Our World: From Hobbies to Possibilities is now complete.

You now understand how to use electronics and code in regular life. Explore, build, and find to learn how technology can make everyday things easier and more fun. You've used your hands and your mind to test circuits, connect wires, and even write some simple code in this lesson. There's more to electronics than just making things work. It can be fun to learn new things and show off your skills.

There are many ways to think, build, and solve problems. From turning on a light bulb to coming up with your own great idea, you can do it your way. Keep looking around, be interested, and don't be afraid to try new things. Every little project you do teaches you something new about technology and how to best use it.

Funded by
the European Union

8. FURTHER READING

If you enjoyed this module and want to learn more, here are some great online sites to continue learning.

1. Science Buddies – Beginner Electronics Projects
Website: https://www.sciencebuddies.org

Simple, step-by-step projects on this website show how electrical circuits and parts work. There are lots of fun and safe projects that you can do there with simple tools.

2. Teach Engineering – Hands-on STEM Activities
Website: https://www.teachengineering.org

This site has projects that have been tried and tested in the classroom and mix technology, creativity, and solving problems. You can find tasks for beginners that link circuits to inventions in the real world.

3. Kodable – Learn Coding through Games
Website: https://www.kodable.com

A bright and fun place to learn how to code where kids can use drag-and-drop to handle cute characters. It helps kids think more logically and teaches them the basics of writing in a fun way.

4. Instructables – Easy DIY Electronics
Website: https://www.instructables.com

A website that is easy to use and full of step-by-step guides on how to make your own small tech projects. You can learn at your own pace with the "Easy Electronics" part, which has clear pictures and short videos.

9. REFERENCE LIST

Excerpt / Idea from Module	Source (APA 7th)	Type
Description of Arduino-based coding activity	Arduino. (n.d.). *Arduino – Open-source electronics platform*. Retrieved from https://www.arduino.cc	Educational tool / coding resource
Reference to MakeCode in coding activity	BBC MakeCode. (n.d.). *MakeCode: Learn coding with fun projects*. Retrieved from https://makecode.microbit.org	Educational coding platform
Image caption: 'DIY Spinner Art STEM Project'	Frog Prince Paperie. (n.d.). *DIY Spinner Art STEM Project*. Retrieved from https://frogprincepaperie.com	Image source
Description of beginner electronics projects for learners	Science Buddies. (n.d.). *Beginner electronics projects*. Retrieved from https://www.sciencebuddies.org	Educational website
Reference to TeachEngineering classroom projects	TeachEngineering. (n.d.). *Hands-on STEM activities*. Retrieved from https://www.teachengineering.org	STEM education resource
Mention of Kodable platform for coding through games	Kodable. (n.d.). *Learn coding through games*. Retrieved from https://www.kodable.com	Educational website
Reference to Instructables for DIY electronics	Instructables. (n.d.). *Easy DIY electronics*. Retrieved from https://www.instructables.com	DIY project platform

9. REFERENCE LIST

Excerpt / Idea from Module	Source (APA 7th)	Type
Generic illustrative images of LEDs, batteries, and circuits	Pixabay. (n.d.). *Free images for education and non-commercial use*. Retrieved from https://pixabay.com	Image repository
Illustrative image examples for classroom materials	Unsplash. (n.d.). *Free educational and technology-related images*. Retrieved from https://unsplash.com	Image repository

Funded by the European Union

10. SELF-ASSESSMENT GRID

Let's see what you remember! Choose the best answer for each question. The correct answer is in **bold**.

Nr.	Question	A	B	C	D
1	What do we call learning by building and exploring electronics with our hands?	Watching videos	Reading about circuits	**Learning by doing**	Copying from others
2	Which everyday item uses electronics?	A pillow	**A washing machine**	A book	A spoon
3	What part gives power to an electronic circuit?	A wire	**A battery**	A bulb	A switch
4	What is the job of a switch?	To make light brighter	**To turn a circuit on or off**	To store energy	To connect batteries
5	Which simple hobby can include both electronics and coding?	**Making a blinking light toy**	Drawing with crayons	Baking a cake	Building with paper
6	What tool helps us to control light or sound using a computer?	A keyboard	**A micro:bit board**	A paper clip	A light bulb
7	In block coding, what do we use to tell the computer what to do?	Sentences	**Colorful code blocks**	Songs	Pictures

Funded by the European Union

Nr.	Question	A	B	C	D
8	What do we call a complete path for electricity to flow?	A circle	**A circuit**	A rope	A wire box
9	What is the purpose of coding in electronic projects like the reminder light, and how does it differ from tinkering?	While tinkering entails providing instructions to the equipment, coding entails physically connecting components.	**While tinkering entails creating circuits using actual components, coding is writing instructions that inform the device what to perform.**	Drawing circuits on paper is a part of coding and tinkering.	Tinkering and coding are equivalent in that they both entail turning a device on and off.
10	What does an LED do in a circuit, and how do you know which side is positive or negative?	Stores electricity; both legs are the same length.	Emits a sound when electricity is applied; the shorter leg has a positive charge.	**Lights up when electricity is applied; the longer leg has a positive charge (+), the shorter leg has a negative charge (–).**	Cools the circuit; the longer leg has a negative charge (–).

Funded by the European Union

MODULE 5: MAKING IT MY OWN – PERSONALISATION & FUN FEEDBACK

Funded by the European Union

Overview

This module highlights the importance of **making learning personal and enjoyable**. For adults with Down syndrome, learning electronics and coding becomes more effective when activities include **choice, creativity, and fun feedback**.

Learners will be encouraged to **adapt projects** according to their preferences (e.g., choosing the colour of LEDs, adding sounds they like, decorating their circuit boards). This sense of ownership increases **motivation, engagement, and pride**.

At the same time, the module emphasises the value of immediate feedback. When pressing a button instantly lights up an LED or when changing a block of code plays a sound, learners understand the connection between action and result. This reinforces confidence and makes learning feel playful rather than stressful.

Finally, the module integrates **positive peer feedback**. Learners will share their projects in a safe and supportive environment, receive encouragement from peers, and learn how to give constructive feedback in simple, easy-to-follow formats.

Simplified language example (Easy-to-Read):

- "I can choose colours and sounds I like."
- "When I press a button, I see or hear something happen."
- "My friends tell me what they liked about my project."
- "I feel proud because my project is mine."

This combination of **personalisation, immediate feedback, and peer recognition** ensures that electronics and coding are not only technical exercises but also tools for **creativity, self-expression, and fun.**

Funded by the European Union

Aim

- To empower learners by giving them the chance to **personalise their projects**.
- To show how **immediate feedback** (lights, sounds, movements) helps in understanding and motivation.
- To encourage learners to give and receive **positive peer feedback** in structured, easy ways.
- To make electronics and coding a space of **fun, creativity, and ownership**.
- To build **confidence and pride** in learners' individual achievements.

Expected Outcomes

By the end of this module, learners will be able to:

1. Adapt a project to reflect their own choices (e.g., colours, patterns, sounds, decorations).

2. Recognise the link between **actions and outcomes** in coding/electronics through immediate feedback.

3. Share their work with peers and respond to positive feedback.

4. Feel more confident and motivated to continue learning because the activity is fun and personal.

Funded by the European Union

1. ADAPT A PROJECT TO REFLECT THEIR OWN CHOICES

One of the main goals of this module is to show learners that technology can be adapted to fit their own ideas and tastes. This is crucial for motivation: when learners feel ownership over a project, they are more engaged, attentive, and proud of the result.

Practical examples of personalisation include:

- **Choosing LED colours**: Each learner can pick the colour that they prefer (e.g., red for energy, blue for calm, green for nature).
- **Customising patterns**: Instead of all lights blinking in the same way, learners can decide if they want fast blinking, slow blinking, or a sequence.
- **Adding decorations**: Learners can use stickers, markers, or small cardboard shapes to make the circuit box look unique.
- **Sound selection**: If the project includes sound, learners can choose a melody or a funny noise they like.

Educational impact: personalisation supports **self-expression**, which is often limited in traditional learning environments. It connects technical learning with the learner's identity and personal preferences.

Simplified language:

- "I choose my colour."
- "I make my project special."
- "It belongs to me."

Practical Activity 1 — My Light & Sound Signature

Purpose
Learners take ownership by choosing how their project looks and sounds. They select colours/patterns for LEDs and a simple sound (beep/buzzer), connect the parts safely, and test for instant feedback. Clear, Easy-to-Read steps and big icons keep the load low.

Duration
~90–120 minutes (demo → pair work → share-out).

Materials
- Battery pack or beginner microcontroller (any board that can blink an LED / beep a buzzer)
- 1–2 LEDs (or a multicolour LED), resistors if needed, small buzzer or speaker
- 1 button or touch pad (foil + wire works)
- 8–10 colour-coded leads
- Laptop/tablet with a block-based editor (large icons) or a safe simulator
- Printed Parameter cards (colour, speed, pattern, sound), Step cards (A5, photo-first)
- Role cards (Builder / Helper)
- Large pictograms (Power / Wire / Test)

Safety 3 (post on the wall)
- **Power OFF** to change wires.
- **Tidy leads** (no loose metal touching battery).
- **Ask before tools**; then test.

Accessibility anchors
- **One idea per line**. Big photos and icons. Short sentences.
- **Immediate sensory feedback** (light/sound) after each step to confirm success.
- **Builder/Helper** roles with a mid-session switch.
- Sentence frames and pictograms for **choice**: “I choose **red**,” “I like **slow blink**,” “I choose **beep**.”

Funded by the European Union

Figures (placeholders to insert later)

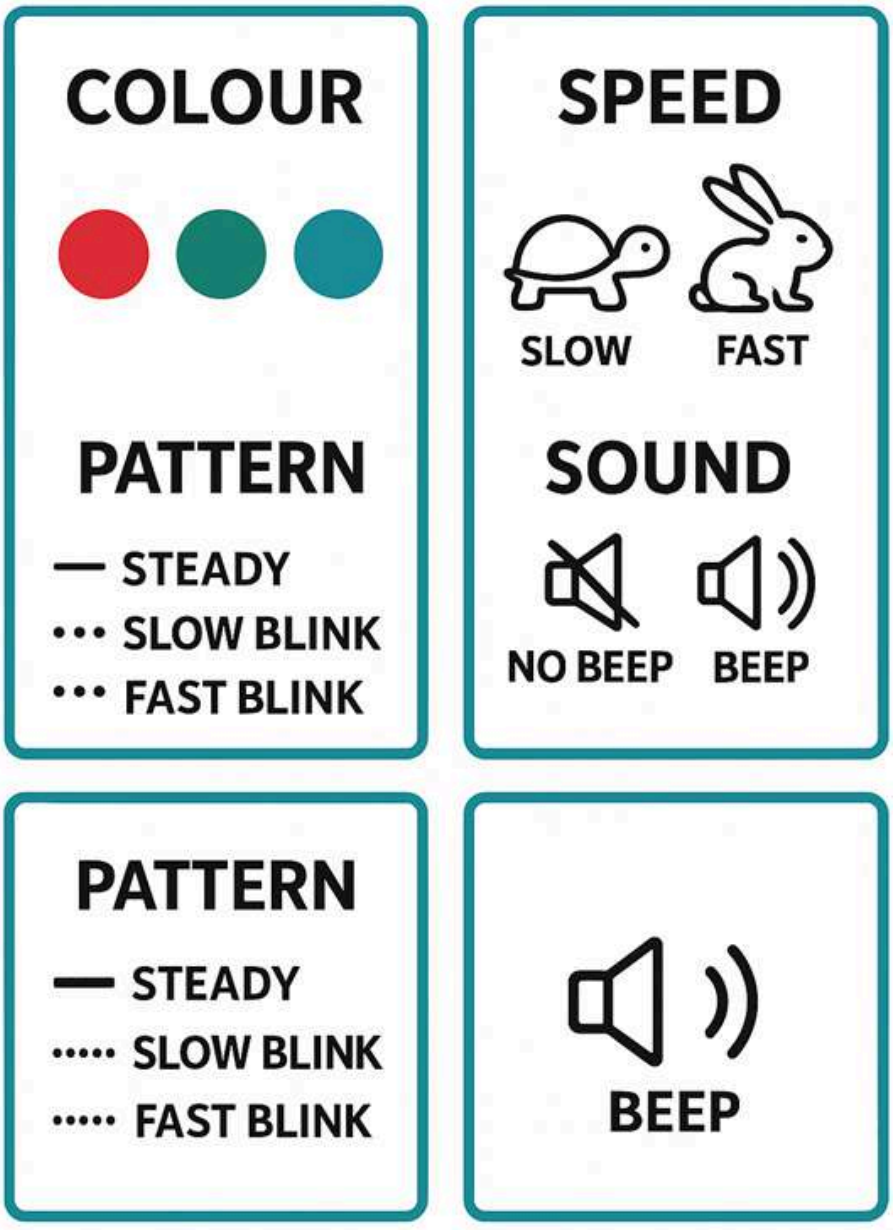


Figure 1. *Parameter cards (colour, speed, pattern, sound).*

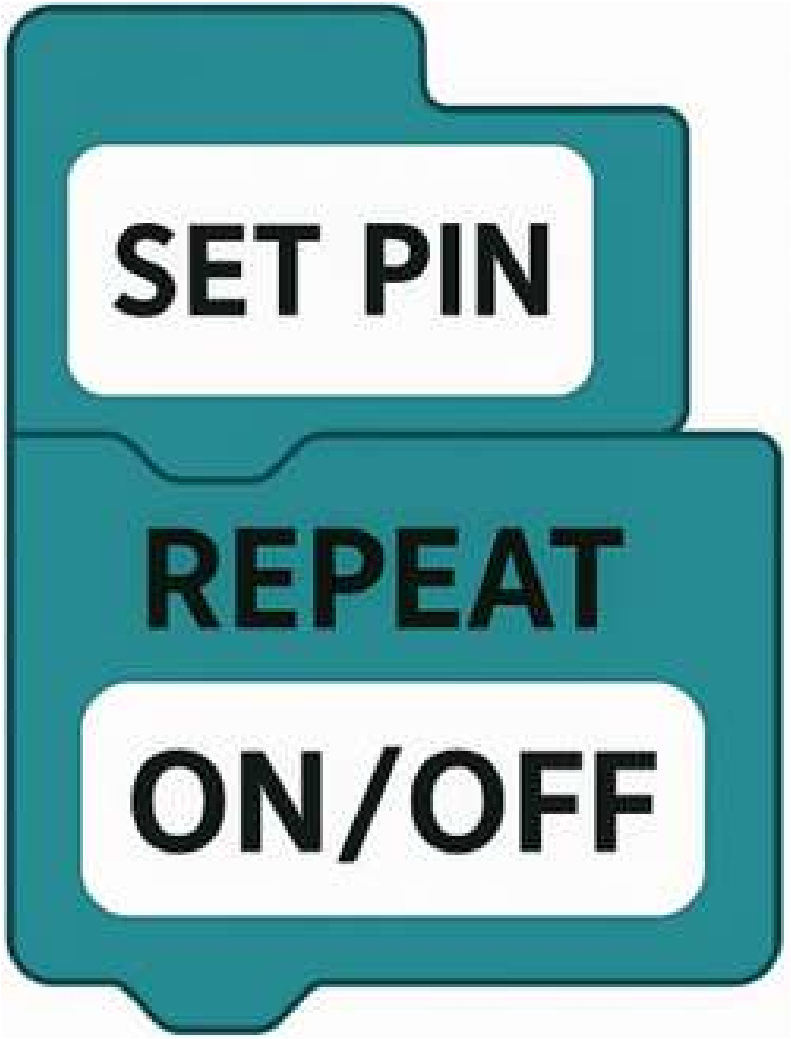


Figure 2. *Block stack example (Set pin → Repeat On/Off).*

Objectives

- Choose **look & feel** (colour/pattern/sound) and say **why I like it**.
- Wire safely and use **3-step debug** if it doesn't work.
- Test and **show** the personalized light/sound.

Funded by
the European Union

Time & grouping

- 10–12 min demo
- 30–40 min pair build & test
- 10–15 min share-out
- 10 min reflection.

Roles: Builder (hands-on) / Helper (reads steps & checks safety). **Switch roles halfway**.

Steps for learners

- **Choose** (point & say):

a) **Colour** (red/green/blue),
b) **Pattern** (fast/slow/steady),
c) **Sound** (beep/no beep).

- **Power OFF**. Connect the parts:

Battery/supply → button/touch pad → LED (+) and buzzer (+).
LED (–) / buzzer (–) → ground/negative.

- If coding: open the **block editor** and add the minimal blocks:

Set pin (LED or buzzer) → **Repeat**: On → Wait → Off → Wait.
Change block options to your **colour/pin** and **speed**.

- **Power ON → Run/Test.** Do you see light / hear sound?
- **Switch roles** and rebuild or tweak one choice (e.g., faster blink or different tone).
- **Photo + caption** (Easy-to-Read): "My style: **blue, slow blink, beep**."

Troubleshooting card — 3-step debug (show as icons)

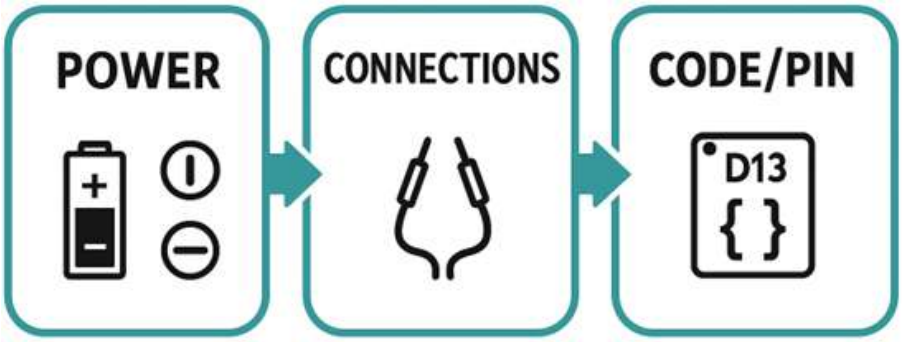


Figure 3. *Debug strip (Power → Connections → Code/Pin).*

Funded by the European Union

A) **Power**: battery in? switch ON? board connected?
B) **Connections**: loose/crossed leads? LED the right way round (+/–)?
C) **Code/Pin**: correct pin selected? wait long enough to see/hear it?

Help-script
"I tried ___. The problem is ___. Can you look at ___?"

Differentiation & AAC (Augmentative and Alternative Communication)

- **Easier**: give a **photo-only** wiring card; use a single LED with one **speed** slider; no sound.
- **Harder**: allow a **multicolour LED**; add **second output** (LED + buzzer); let learners design a **two-step pattern** (e.g., long–short).
- **AAC**: symbol cards for **Choose / Connect / Test / Like**; colour swatches to point at.

Facilitators actions

- Keep choices **concrete**: point to a card, say it, then implement.
- After any change, ask: "What **changed**? What do you **see/hear** now?"
- When stuck, **gesture** the 3-step debug instead of re-explaining (tap Power icon → follow wire with finger → point to Code/Pin).

Mini-checks (during activity)

- "Show me your **input** and your **outputs** with your finger."
- "Say your **one-sentence** behaviour: 'When I press, **it blinks blue and beeps**.'"

Evidence & observation checklist (tick per pair)
□ Took **roles** and **switched** once
□ Followed **Power-OFF** rule while rewiring
□ Project **reacted** (light/sound) with chosen settings
□ Used **3-step debug** before asking for help
□ Gave a **photo + one-line caption** of their style

Funded by the European Union

Quick share-out (10–15 min)
• Pairs place their gadget on the table. A neighbour **presses** the button/touch pad and **describes** what happened in one line.
• Host says: "We chose ___ because ___."
• Visitors place + (Two stars & a wish). Hosts **log one idea** on a strip.

Reflection (exit ticket; Easy-to-Read)
"One thing I **personalised**: _____."
"One idea for **next time**: _____."

Prep for Practical Activity 2 (what comes next)
• Keep your **Parameter card** (colour/pattern/sound).
• In the next practical activity we add a tiny **game rule**: "After **3 successes → celebration** (your light & sound show)."

Funded by the European Union

2. RECOGNISE THE LINK BETWEEN ACTIONS AND OUTCOMES IN CODING/ELECTRONICS THROUGH IMMEDIATE FEEDBACK

Immediate feedback is one of the most effective learning tools for learners with intellectual disabilities, because it shortens the distance between action and result. When something happens right away, learners can understand the cause-and-effect relationship.

Examples:

- **Pressing a button → LED lights up.**
- **Changing a block of code → the sound changes immediately.**
- **Connecting a sensor → the motor starts spinning.**

This feedback loop allows learners to:

- **Confirm success**: "Yes, it works."
- **Notice errors quickly**: "Nothing happened → maybe I forgot a wire."
- **Celebrate small wins**: Every blink or beep is a reason to feel proud.

Educational impact: feedback builds confidence and reduces frustration. Learners see proof that their actions matter and that they are capable of controlling technology. It also encourages problem-solving — when something doesn't work, the learner knows to check and try again.

Simplified language:

- "I press → I see light."
- "I change → I hear sound."
- "If it does not work, I try again."

Practical Activity 2 — Reaction Glow

Purpose
A simple attention game that strengthens cause→effect thinking and timing. Learners press **when the LED turns on; faster reaction** earns a **longer celebration**.

Materials
- Their practical activity 1 gadget (button or **touch pad** + LED(s) + buzzer)
- Laptop/tablet with **block-based editor** (large icons) or safe **simulator**
- Printed **Flow strip** ("Touch → +1 → if 3 → Celebration") and **Block cue card** (counter, if/then, reset)
- **Score tokens** (three circles) • Step cards (A5, photo-first)
- Help-script & Debug strip cards

Steps (A5 card)
10. Program a **random wait** (1–3 s), then **LED ON**.
11. Learner **presses** as soon as the LED is ON.
12. If **pressed quickly** → play **long celebration**; else → play **short celebration**.
13. **Reset** and try again. Keep **two scores** (smiley tokens): (fast) | (okay).

Differentiation
- **Easier**: remove timing; any press after LED ON = celebration.
- **Harder**: log **best time**; allow **two levels** of celebration.

Observation checklist (tick per pair)
□ Counter works (0→1→2→3→**celebration**→0)
□ Learner **said the rule** once
□ Debug used **before** asking for help
□ Celebration is **visible/audible**
□ Peer feedback given (+)

Funded by the European Union

Team routines — mini-gallery game (10–15 min)

- Each pair hosts a "**3 touches = celebration**" station. Visitors press and watch.
- Visitors place + and say: "I liked ___; maybe next time ___."
- Hosts **log one idea** on their mat.

Reflection (exit ticket; Easy-to-Read)

"Our **rule** was: _____."
"We will **change** next time: _____."

3. FEEL MORE CONFIDENT AND MOTIVATED TO CONTINUE LEARNING BECAUSE THE ACTIVITY IS FUN AND PERSONAL

The ultimate outcome of this module is not just technical knowledge, but an increased sense of **confidence and motivation**. By combining personalisation, immediate feedback, and positive peer interaction, learners discover that learning can be fun and meaningful.

Practical signs of confidence:

- Learners smile and show pride in their projects.
- Learners explain their project to others without fear.
- Learners volunteer to try new tasks.

Long-term impact: learners are more likely to want to continue learning after the project ends — at home, with friends, or in new courses. Enjoyment fuels curiosity, and curiosity leads to lifelong learning.

Simplified language:

- “I am proud.”
- “Learning is fun.”
- “I want to learn more.”

Funded by the European Union

EVALUATION & SELF-ASSESSMENT

Let's see what you remember! Choose the best answer for each question. The correct answer is in **bold**.

Nr.	Question	A	B	C	D
1	When building your project, what is the safest first step?	Connect wires randomly	Test before turning power OFF	**Turn power OFF before changing wires**	Ignore loose cables
2	Personalising your project means…	Making it look exactly like your partner's	**Choosing colours, patterns, or sounds you like**	Using only one LED	Following all instructions without changes
3	Immediate feedback happens when…	You wait a few minutes to see results	You need the teacher to tell you if it worked	**You press a button and see or hear something right away**	You record the result for later
4	In the "Reaction Glow" activity, what should you do?	Press the button at any time	**Press the button when the LED turns ON**	Wait until the buzzer beeps twice	Turn the LED OFF manually
5	What is the first step of the 3-step debug routine?	Edit the code blocks	Change LED colour	**Check the power and connections**	Ask your partner to fix it
6	What helps you stay motivated during the module?	Competing to win every task	**Having fun and seeing your own progress**	Copying other projects	Finishing as fast as possible
7	Good peer feedback looks like…	"It's bad."	"You did everything wrong."	**Two positives and one idea for improvement**	No comment

Funded by the European Union

Nr.	Question	A	B	C	D
8	Which sentence is a clear caption for your project?	“Beep.”	**“Blue slow blink because I like calm lights.”**	“I made something.”	“My project is good.”
9	When something doesn’t work, what should you do first?	**Try the 3-step debug**	Call the facilitator immediately	Remove all parts and start again	Wait for someone else to fix it
10	Feeling confident after this module means…	**Being proud and wanting to learn more**	Never making mistakes again	Working alone all the time	Avoiding new challenges

Funded by the European Union

FURTHER MATERIAL

Easy-to-Read standards & style (EN/DE)

- Inclusion Europe — Information for all: European standards for easy-to-read materials (multi-language set + "How to use" notes). Great for wording captions and instructions. Inclusion Europe
- Netzwerk Leichte Sprache — Die Regeln für Leichte Sprache (Neuauflage 2022) (PDF). Use for German learner-facing text. netzwerk-leichte-sprache.de
- IFLA Professional Report 120 — Guidelines for Easy-to-Read Materials. Clear, international baseline for accessible text layout. ifla.org
- Bundesfachstelle Barrierefreiheit — overview page linking to Leichte-Sprache rules and federal guidance (DE). Bundesfachstelle Barrierefreiheit

UDL & inclusive design

- CAST — UDL Guidelines 3.0 (interactive). Apply "Provide multiple means of action & expression" to our photo/caption/voice choices. udlguidelines.cast.org

Tinkering & block coding (blink / beep basics)

- Arduino — Blink and BlinkWithoutDelay (official docs). Good reference if you extend beyond blocks. arduino.cc
- micro:bit — Get started and MakeCode: Getting started (block editor). Ideal for press→blink-beep patterns. microbit.org
- Tinkercad Circuits — LED blink tutorials and Learn hub (simulator for practice without hardware). Tinkercad

Peer feedback ("Two stars & a wish")

- NWESD one-pager (PDF) — classroom routine adapted from Dylan Wiliam's formative assessment; perfect to brief helpers. NWESD 189
- Cambridge University Press & Assessment — formative assessment kit with "Two stars & a wish" template. Cambridge English
- K20 Center strategy card — concise, adult-friendly explanation. learn.k20center.ou.edu

Funded by the European Union

Symbol & pictogram resources for Easy-to-Read artefacts

- ARASAAC — large, free AAC pictogram library (check CC-BY-NC-SA). beta.arasaac.org
- OpenSymbols — open-licensed collection (>50k) spanning several sets. opensymbols.org

Widgit — widely used commercial symbol set; see licensing if you plan to publish. Widgit Software

Down syndrome — education & adult life (context for facilitators)

- Down Syndrome Education International (DSE) — evidence-based resources and recommendations for education. down-syndrome.org
- European Down Syndrome Association (EDSA) — EU-wide network, webinars, and practice-sharing. edsa.eu
- Adult Down Syndrome Center — practical activity ideas for adults (for enrichment days around the module). adscresources.advocatehealth.com

Funded by the European Union

REFERENCES

- Arduino. (2024, February 10). Blink. https://www.arduino.cc/en/Tutorial/Blink
- Autodesk. (n.d.). Learn Circuits. https://www.tinkercad.com/learn/circuits
- Cambridge University Press & Assessment. (2022). Formative assessment kit (Resource #3: Two stars and a wish) [PDF]. Cambridge English
- CAST. (2024). UDL Guidelines 3.0. https://udlguidelines.cast.org
- Down Syndrome Education International. (n.d.). Down Syndrome Education Online (resources). https://www.down-syndrome.org
- Down Syndrome Education International. (n.d.). Recommendations for the education of learners with Down syndrome. ds-int.org
- European Down Syndrome Association. (n.d.). EDSA — Association website & webinars. edsa.eu
- IFLA. (2010). Guidelines for easy-to-read materials (IFLA Professional Reports, No. 120). International Federation of Library Associations and Institutions. ifla.org
- Inclusion Europe. (2021, October 6). Information for all: European standards for making information easy to read and understand. Inclusion Europe
- Inclusion Europe. (2023, December 10). How to use the European standards (FAQ). inclusion.eu
- Micro:bit Educational Foundation. (n.d.). Getting started (Introduction / Get coding). https://microbit.org/get-started/getting-started/introduction/ and https://microbit.org/get-started/getting-started/get-coding
- NWESD. (2011). Two Stars and a Wish [PDF]. NWESD
- OpenAAC Initiative. (n.d.). OpenSymbols. opensymbols.org
- Netzwerk Leichte Sprache e.V. (2022). Die Regeln für Leichte Sprache (Neuauflage) [PDF]. netzwerk-leichte-sprache.de
- Adult Down Syndrome Center. (2023, December). Activities you can do at home. adscresources.advocatehealth.com
- Widgit Software. (n.d.). Widgit Symbols — Licensing & copyright guidance. https://www.widgit.com/symbol-services/licensing.htm and https://www.widgit.com/symbol-services/copyright.htm
- ARASAAC. (n.d.). AAC pictograms (open-access symbol set). beta.arasaac.org
- (If you prefer the MakeCode link directly: Microsoft. (n.d.). MakeCode for micro:bit — Getting started.*

Funded by the European Union

DISLAIMER

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or State Education Development Agency (SEDA). Neither the European Union nor the granting authority (SEDA) can be held responsible for them.