

# LEGO® Education WeDo 2.0

## Computing Extension Projects



WeDo 2.0  
2045301

# Table of contents

**WeDo 2.0 Computing  
Extension Projects**

**3-14**

**WeDo 2.0 in the Computing  
Curriculum**

**15-19**

**Assess Computing with  
WeDo 2.0**

**20-27**

**Experimentation Activity**

**28-35**

**Guided Projects Extensions**

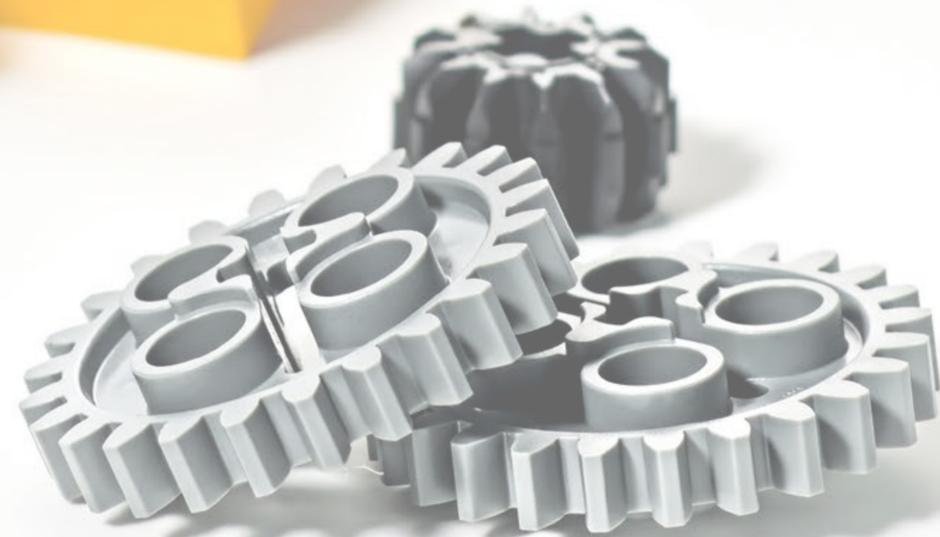
**36-94**

**Open Projects Extensions**

**95-109**

**WeDo 2.0 Toolbox**

**110-121**



# WeDo 2.0 Computing Extension Projects

Welcome to the LEGO® Education WeDo 2.0 Computing Extension Projects.

In this chapter, you will discover how you can use this WeDo 2.0 solution to teach basic programming skills in a science context.





## LEGO® Education WeDo 2.0 Computing Extension Projects

LEGO® Education is pleased to present computing project for use in primary education.

These materials will help you to deliver exciting projects based around relevant technology, which will enable pupils to learn fundamental computing and programming concepts using a wider curriculum and real-world context, while expanding on the projects found in the LEGO Education WeDo 2.0 Curriculum Pack.

WeDo 2.0 supports a hands-on, minds-on learning solution that gives pupils the confidence to ask questions, the tools to find the answers, and the skills required to solve real-life problems.

Computing is an increasingly important component of the lives of pupils. Understanding how machines and computers process information is a valuable skill to develop at a young age. It is important to encourage pupils to experiment with logical ideas, and allow them to explore and question what they do not yet understand.

Computing in WeDo 2.0 brings the pupils' creations to life, generating excitement and the desire to discover more.





## Develop programming skills through projects

These computing projects are extensions to the science projects already developed in the LEGO® Education WeDo 2.0 Curriculum Pack.

To this end, the Computing Curriculum Purpose of Study will be addressed:

*“A high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world. Computing has deep links with mathematics, science and design and technology, and provides insights into both natural and artificial systems. The core of computing is computer science, in which pupils are taught the principles of information and computation, how digital systems work and how to put this knowledge to use through programming. Building on this knowledge and understanding, pupils are equipped to use information technology to create programs, systems and a range of content. Computing also ensures that pupils become digitally literate – able to use, and express themselves and develop their ideas through, information and communication technology – at a level suitable for the future workplace and as active participants in a digital world.”*

This document consists of one activity and eight Computing Extension Projects:

- One Experimentation Activity to quickly explore programming concepts.
- Six project extensions to Guided Projects.
- Two project extensions to Open Projects.

Each Computing Extension Project could take between one and three lessons of approximately 45 minutes, depending on how you organise them. There are no references to the computing extensions in the software. All of the information is provided in this document.

### Reference

National Curriculum in England : computing programmes of study : key stages 1 and 2, Department for Education.



## Target Group

The material is aimed at Key Stage 2 but can be adapted for any primary school year group. Whilst the material has primarily been written to address the computing curriculum, the projects draw on the science and geography concepts introduced in the LEGO® Education WeDo 2.0 Curriculum Pack, and the focus has been directed to investigating, modelling, and designing using programming.

These computing Extension Projects are designed to develop pupil understanding and programming skills, and to demonstrate the importance of computational thinking and the effective use of technology in everyday life. Pupils will gain experience in basic programming through a combination of direct teaching, experimentation, and exploration. These ample cross-curricular opportunities mean that the Extension Projects can be delivered contextually and will impact on other subject areas.

Important elements to consider before you start:

- The teacher's guide and resources will reduce planning time, and the experience gained will help you when making further plans.
- Each Extension Project is composed of different tasks that can be used as a differentiation tool.
- Allow each pupil to follow the progression route, as they will progressively develop their programming skills.
- Specific skills and concepts are addressed in the teacher's guide provided for each Extension Project.
- The sharing aspect will also impact on pupils' oral communication skills.





## Organisation of the Extension Projects

The Extension Projects presented in this document can be delivered as stand-alone computing lessons, but they will have the most impact when delivered contextually, extending the science projects from which they are developed. The Guided Projects in the LEGO® Education WeDo 2.0 Curriculum Pack have three phases: Explore, Create, and Share. The best time to introduce an Extension Project is at the end of the Create phase. At that stage, the pupils will have built a model and will know the behaviour of it. You can use the Explore scenario proposed in this document to start the Computing Extension Projects.

### About building

In most cases, the LEGO model presented in the software will not need adjustment, but if so, it will only need a few adjustments in order to work. Building inspiration has been provided in this document to help you with guiding pupils when implementing any changes.

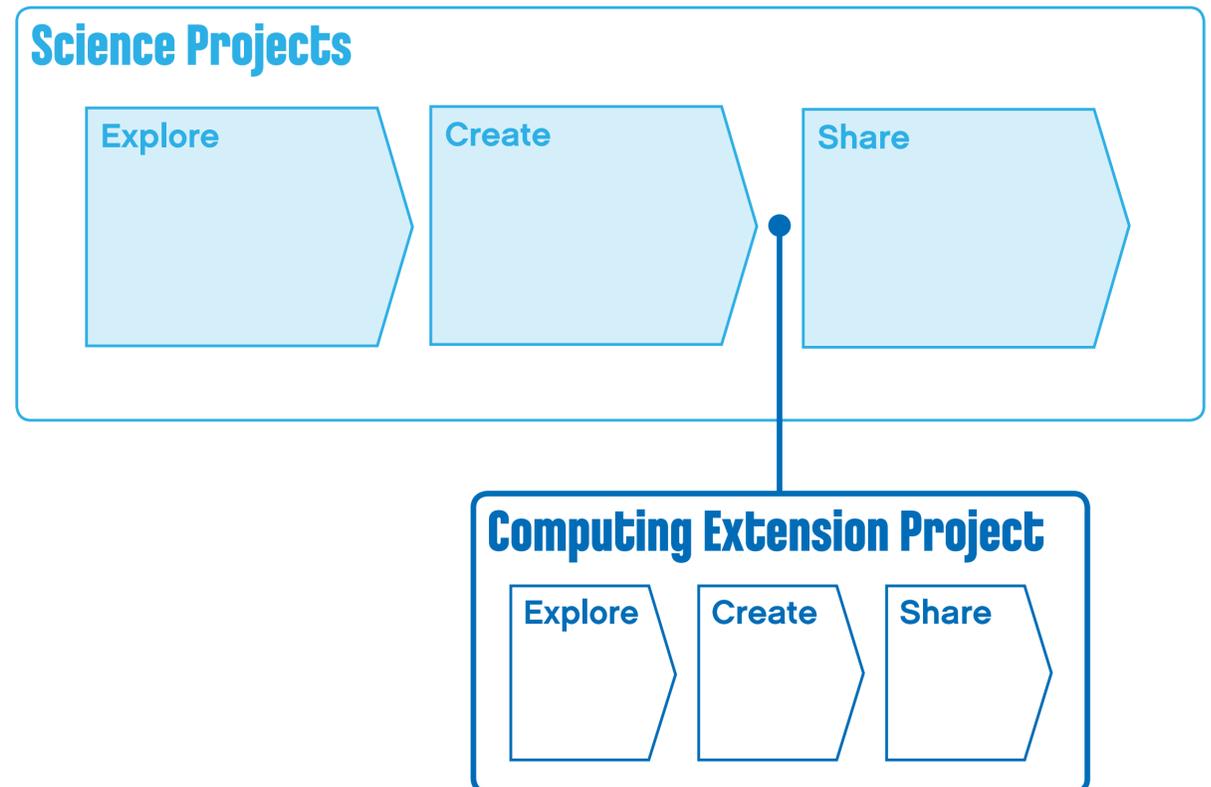
In some cases, pupils could spend more time rebuilding their models to develop better solutions to the problems.

### About programming

In all cases, we have provided examples of programs to be used as guidelines. In general, it is important for pupils to develop their own programs, as there are multiple possible solutions to each problem.

### ▶ Important

During each phase, pupils will document their findings, the answers, and the process, using various methods. The digital document they produce can be exported and used for assessment, display, or sharing with parents.





## Things to consider

### About the hardware

All of the bricks the pupils need for the sessions can be found in the LEGO® Education WeDo 2.0 Core Set. It is important to point out expectations before using the set with the class. For example, the pupils should make sure that the sets are put away in the same condition in which they were received.

### About the software

The LEGO Education WeDo 2.0 software will be used in these Extension Projects. It is recommended that you familiarise yourself with the software before introducing it to the pupils. The Design Library and the Project Library referred to in this document are available in the full version of the software.

### Vocabulary

Each project has been designed in a way that will enable you and your pupils to acquire an increasingly accurate vocabulary. Some pupils may already be familiar with the technical terms, but it is important that all pupils are encouraged to use the correct terms as quickly and as frequently as possible.

You can find a glossary of the computing terms used in these projects in the Toolbox chapter, along with child-friendly definitions to share with the class.





## More things to consider

### Use of the Internet, including video hosting websites

No direct links to search engines or video hosting websites are provided, as they can become obsolete over time. To overcome this restriction, a number of keywords have been provided for use with your preferred search engine when sourcing appropriate content.

### Solution Examples

Examples of possible solutions for each project stage are provided in the teacher's guide. While there may be occasions when it is necessary to provide your pupils with these solutions (i.e., for differentiation purposes), it is recommended that pupils are encouraged to find their own solutions. Whenever a solution is given, it is advisable to point out to the pupils that it is only one solution, and that, in most cases, other solutions can be found.

### Important

The example solutions provided were designed with a specific model in mind. Although the set-up procedure has been described as accurately as possible, bear in mind that you may have to adjust the program strings to fit your needs.



## Carrying out the Experimentation Activity

The Experimentation Activity takes the form of an open-ended problem-solving task. Pupils will become familiar with the WeDo 2.0 LEGO® elements, and develop an understanding of the programming software. There will be opportunities for them to record and share their work using the built-in facilities of the software.

The focus of this activity is on exciting and engaging the pupils, and on communicating the fact that everyone can program a motor to move and a light to blink. Let the pupils discover a wide range of solutions to the task. Each different solution should be celebrated as being correct and unique.

### ► Suggestion

In addition to the Experimentation Activity, the Getting Started Project “Milo the Science Rover” can be used to introduce pupils to WeDo 2.0. This project has been designed to guide pupils through most of the possibilities of the bricks and software. By completing the four parts of this Getting Started Project, pupils will gain experience with:

- Building to plan
- Connecting the Smarthub to their devices
- Controlling the motor through simple programming
- Controlling the motion sensor through simple programming
- Controlling the tilt sensor through simple programming
- Using different outputs such as light, sound, and images through simple programming
- Documenting their work





## Using the Guided Projects extensions

There are six Guided Project computing extensions contained in this document:

They are:

- Speed Control (an extension of Speed)
- Dancing Bees (an extension of Plants and Pollinators)
- Debug the Floodgate (an extension of Prevent Flooding)
- Fearless Frog (an extension of the Frog's Metamorphosis and Predator and Prey 'Open Projects')
- Rescue Count (an extension of Drop and Rescue)
- Reverse and Recycle (an extension of Sort to Recycle)

They use the same basic models and themes as the original projects, but the focus moves to developing programming skills through further modelling or designing.

Each Extension Project contains two to three tasks that gradually increase in difficulty and complexity. They can be undertaken together or separately, thereby offering opportunities for differentiation.

While pupils should be encouraged to find and record their own solutions to the tasks, the project extension chapters do contain suggested programming solutions and concepts to focus on. Where models need to be modified, detailed images of suggested modifications are provided.





## Using the Open Project Extensions

There are two Open Project computing extensions contained in this document:

- Smart Lift (an extension of Moving Materials)
- Working Rover (an extension of Space Exploration)

These Extension Projects follow the same structure as the Guided Project extensions, but are more “open”, thus providing customisation opportunities.

Suggestions for programming solutions are not provided in these project extension chapters, as each pupil is expected to design and build their own unique model. However, links to the Design Library of the LEGO® Education WeDo 2.0 Software have been provided. Here, pupils will find inspiration for creating models with particular functions.





## Document the projects

Asking your pupils to document their work will help you to keep track, identify where they need more help, and evaluate their progress.

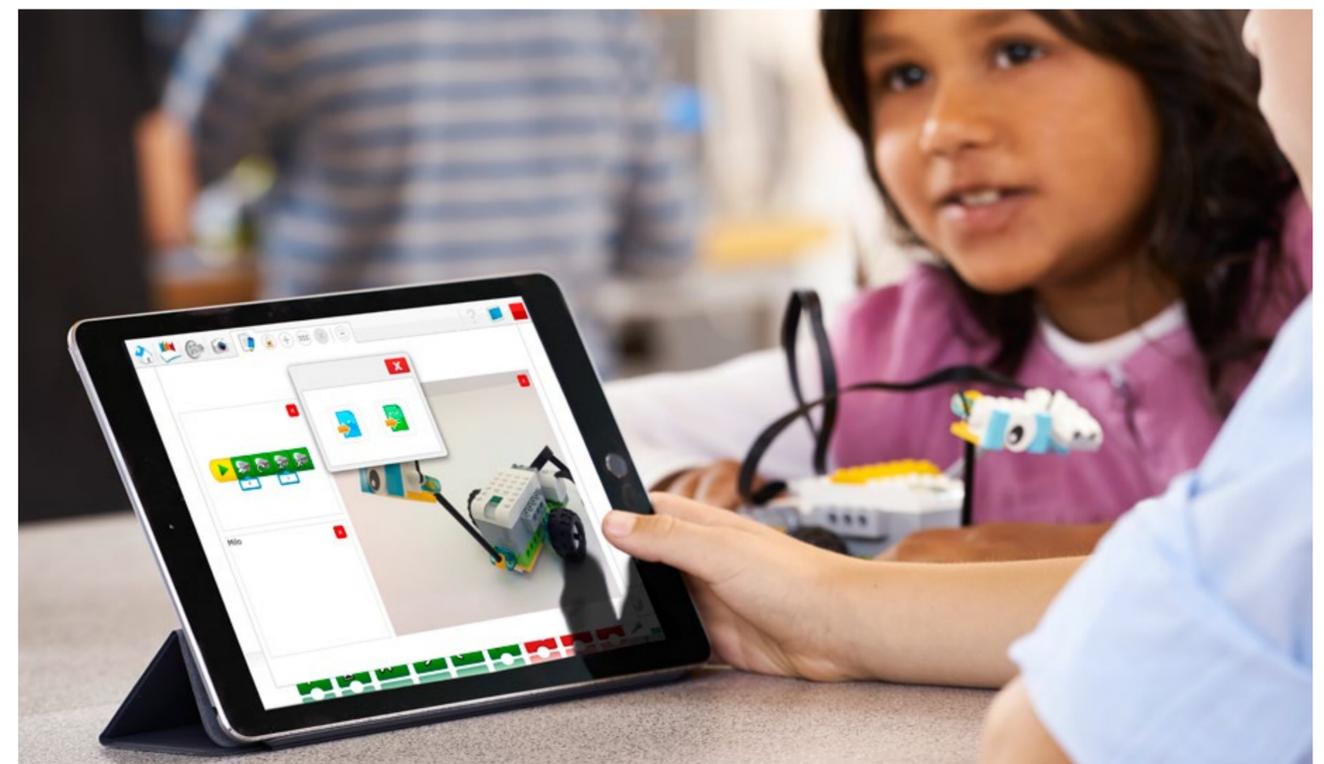
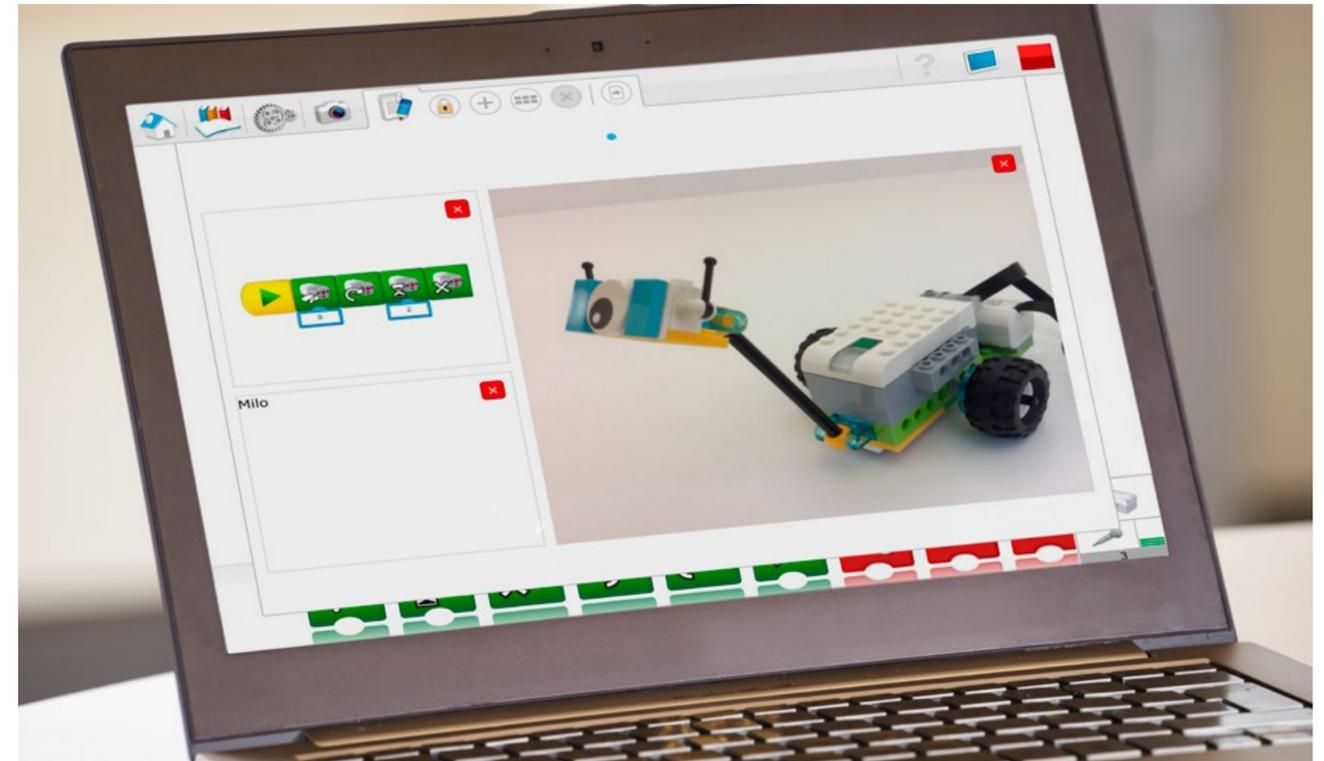
Pupils can use many different methods to express their ideas. During the ongoing documentation process, they can:

1. Take photographs of important steps of their prototypes and their final models.
2. Take photographs of their team working on important stages of the process.
3. Record a video explaining a problem they are facing.
4. Record a video explaining their investigation.
5. Make notes using the Documentation tool.
6. Find supporting pictures on the Internet.
7. Take screenshots of their programs.
8. Write, draw, or sketch on paper and then take photographs to record the information.

The LEGO® Education WeDo 2.0 software features a built-in documentation tool, where pupils can document their work without the need for additional applications, creating a more fluid and integrated experience.

### ▶ Suggestion

A combination of paper and digital documentation can be the most effective, depending on the age group you are working with.





## Share the projects

At the end of each project, pupils will be eager to share their programming solutions and findings. This is a great opportunity to develop their communication abilities.

Here are a few examples of how your pupils can share their work:

1. Ask the pupils to create the display where the LEGO® model will be used with a range of different programming solutions.
2. Ask a team of pupils to present their best solution to you, another team, or to the class.
3. Invite an expert or a group of parents to your classroom for a pupil presentation.
4. Ask the pupils to record videos explaining their projects, and post them online.
5. Create and display posters of the projects around your school.
6. Email the project documents to parents, or publish them in the pupils' portfolios.

### ▶ Suggestion

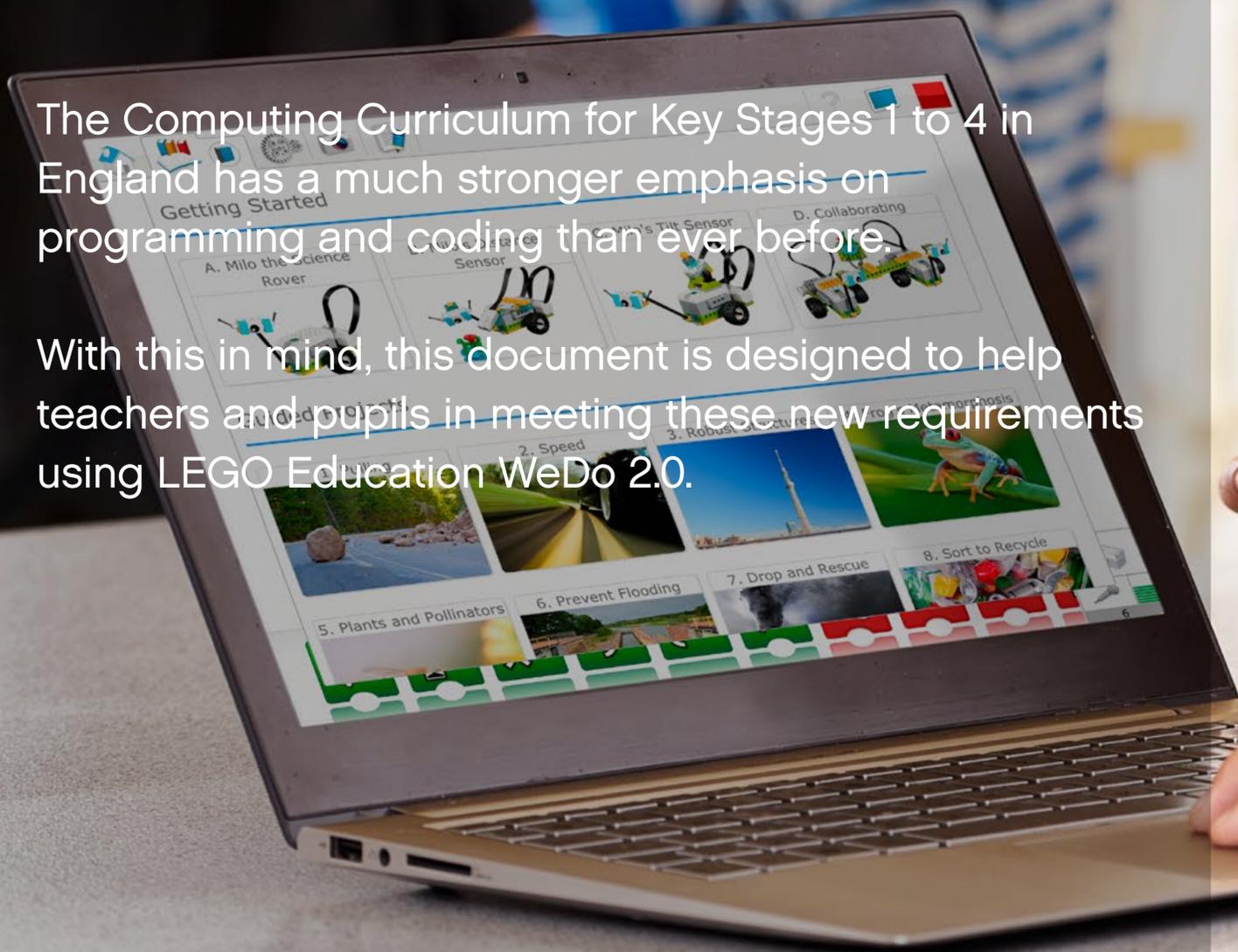
To make this experience even more positive, ask each pupil to make a positive comment or to pose a question about another pupil's work during the sharing session.



# WeDo 2.0 in Computing Curriculum

The Computing Curriculum for Key Stages 1 to 4 in England has a much stronger emphasis on programming and coding than ever before.

With this in mind, this document is designed to help teachers and pupils in meeting these new requirements using LEGO Education WeDo 2.0.





## Computing Curriculum Links

This material is aimed Key Stage 2 but can be adapted for any primary school year group.

The National Curriculum in England Computing Programme of Study aims to ensure that all pupils:

- can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms, and data representation.
- can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems.
- can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems.
- are responsible, competent, confident, and creative users of information and communication technology.

To reach those goals, the LEGO® Education Computing Extension Projects have developed direct links to the requirements of the Computing Curriculum at Key Stage 2.

Using these materials, pupils will develop their skills and an understanding of these requirements.

Pupils should be taught to:

- design, write, and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts.
- use sequence, selection, and repetition in programs; work with variables and various forms of input and output.
- use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and problems.
- use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content.
- select, use, and combine a variety of software (including Internet services) on a range of digital devices to design and create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information.
- use technology safely, respectfully and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact.
- understand computer networks including the Internet; how they can provide multiple services, such as the World Wide Web; and the opportunities they offer for communication and collaboration.



## Use LEGO® bricks in a computational-thinking context

Computational thinking is a set of problem-solving skills that are developed while working with computers and other digital devices. In WeDo 2.0, computational thinking is developed by combining LEGO® builds with the use of simple programming strings. Pupils can activate motors, lights, sounds, and displays, or they can program their models and prototypes to react to sound, tilt, or movement to implement functions.

Programming with WeDo 2.0 is handled in a developmentally appropriate manner through the use of textless programming blocks. These blocks have been designed to provide rich and simple programming opportunities for pupils.

These opportunities include:

- Logical reasoning
- Ordering actions
- Looking for patterns
- Developing algorithms
- Organising and analysing data
- Using models and simulations
- Using computers to collect and report information

The use of digital tools in science and engineering projects encourages computational thinking, which enables pupils to carry out investigations, develop models, and design solutions to problems more effectively.

The excitement of seeing their own creations come to life motivates pupils and encourages them to develop new skills.





## Visual overview of the Extension Projects

### 1. Speed Control

This project is about designing programs that will help a race car driver to determine and control the speed of a race car.

### 2. Fearless Frog

This project is about modelling a frog that can recognise, and escape from, dangerous predators.

### 3. Dancing Bees

This project is about modelling the dance of a bee that communicates the distance and location of pollen-bearing flowers, and how to return to the hive after foraging.

### 4. Debug the Floodgate

This project is about designing programs to correct errors in the way a floodgate is working.

### 5. Rescue Count

This project is about designing programs that can help with counting the number of rescued animals.

### 6. Reverse and Recycle

This project is about designing programs to make a recycling truck safer for the driver and pedestrians.

### 7. Smart Lift

This project is about designing programs that will help with the atomisation of a factory warehouse.

### 8. Working Rover

This project is about collaborating with other groups to design and develop a multifunctional space rover for collecting soil samples.





# Overview of the Extension Projects organised by National Curriculum requirements

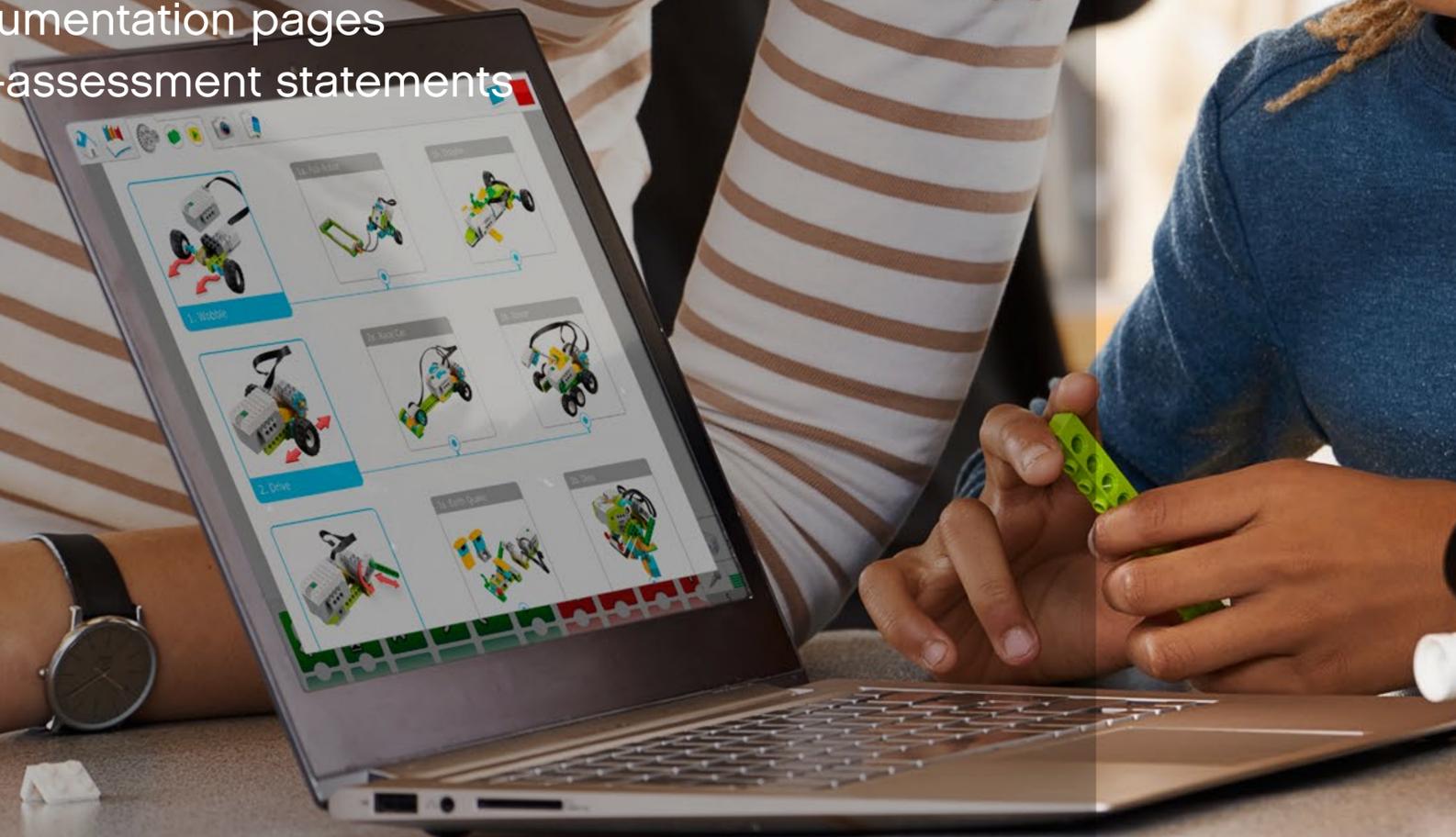
Learning Outcome/Curriculum Link	1	2	3	4	5	6	7	8
Pupils should be taught to:	Speed Control	Fearless Frog	Dancing Bees	Debug the Floodgate	Rescue Count	Reverse and Recycle	Smart Lift	Working Rover
Design programs that accomplish specific goals	●	●	●	●	●	●	●	●
Write programs that accomplish specific goals	●	●	●	●	●	●	●	●
Use sequence in programs			●	●	●	●	●	
Work with various forms of input	●	●	●	●	●	●	●	●
Work with various forms of output	●	●	●	●	●	●	●	●
Debug programs that accomplish specific goals				●		●	●	●
Use repetition in programs	●	●	●	●	●	●	●	●
Control or simulate physical systems	●			●	●	●	●	●
Solve problems by decomposing them into smaller parts	●			●	●			●
Use selection in programs	●	●	●	●	●	●	●	●
Work with variables	●					●	●	●
Use logical reasoning to explain how some simple algorithms work	●	●	●		●	●	●	●
Use logical reasoning to detect and correct errors in algorithms				●		●	●	
Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information	●	●	●	●	●	●	●	●
Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content			●				●	●
Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact	●	●	●	●	●	●	●	●

● = Main learning outcome/curriculum link   ● = Supporting learning outcome/curriculum link

# Assess with WeDo 2.0

There are many ways to monitor and assess your pupils' progress through a WeDo 2.0 project. With this in mind, the following assessment tools are provided:

- Anecdotal record grid
- Observation rubrics grid
- Documentation pages
- Self-assessment statements





## Teacher-led assessment

Developing pupils' computational thinking takes time and feedback. Just as in the design cycle, in which pupils should know that failure is part of the process, assessment should provide feedback to pupils in terms of what they did well and where they can improve.

Problem-based learning is not about succeeding or failing. It is about being an active learner and continually testing and building upon ideas.

### **Anecdotal record grid**

The anecdotal record grid lets you record any type of observation you believe is important about each pupil. Use the template on the next page to provide feedback to pupils about their learning progress as required.



# Anecdotal record grid

Name: \_\_\_\_\_ Class: \_\_\_\_\_ Project: \_\_\_\_\_

Emerging	Developing	Proficient	Accomplished

Notes:



## Teacher-led assessment

### Observation rubrics

An example rubrics has been provided for every Guided Project. You can use the observation rubrics grid to:

- Evaluate pupil performance at each step of the process.
- Help pupils to progress by providing constructive feedback.

Observation rubrics provided in the Guided Projects can be adapted to fit your needs. The rubrics are based on these progressive stages:

#### 1. Emerging

The pupil is at the beginning stages of development in terms of content knowledge, ability to understand and apply content, and/or demonstration of coherent thoughts about a given topic.

#### 2. Developing

The pupil is able to present basic knowledge only (vocabulary, for example), and cannot yet apply content knowledge or demonstrate understanding of concepts being presented.

#### 3. Proficient

The pupil has concrete levels of understanding of content and concepts and can demonstrate adequately the topics, content, or concepts being taught. The ability to discuss and apply outside the required assignment is lacking.

#### 4. Accomplished

The pupil can take concepts and ideas to the next level, apply concepts to other situations, and synthesise, apply, and extend knowledge to discussions that include extensions of ideas.

### Suggestion

You can use the observation rubrics grid on the next page to keep track of your pupils' progress.



## Computing Assessment Rubrics

You can use these assessment rubrics with the observation rubrics grid, found on the next page.

### Explore Phase

During the Explore phase, make sure each pupil is actively involved in the discussion, asks and answers questions to identify the question or the problem they have to solve, and correctly uses computing vocabulary.

1. The pupil is unable to provide answers to questions, participate in discussion or adequately use computing vocabulary.
2. The pupils is able, with prompting, to adequately provide answers to questions or participate in discussion, or with help, use computing vocabulary.
3. The pupil is able to provide adequate answers to questions, participate in class discussion, or use computing vocabulary.
4. The pupil is able to extend explanations and discuss or describe using computing vocabulary very precisely.

### Create Phase

During the Create phase, make sure each pupil is working as part of a team, that they explore programming solutions and that they can use information gathered during the Explore phase.

1. The pupil is unable to work as part of a team, explore different programming solutions, or use gathered information.
2. The pupil is able to work as part of a team and, with help, is able to explore a small range of programming solutions.
3. The pupil is able, with guidance, to gather and use information, work as part of a team, contribute to team discussion, explore a wider range of programming solutions, and gather information to use in a presentation.
4. The pupil is able to work as part of a team, serve as a leader, use gathered information to inform their programming solutions and explain them.

### Share Phase

During the Share phase, make sure that each pupil can explain what is happening with their model in relation to their programming solution, and can use information from the project to create a final report.

1. The pupil is unable to engage in the discussion about the project, is unable to explain the programming solution, or unable to use the information to create a final project.
2. The pupil is able, with prompting, to engage in a discussion about programming solutions, complete testing scenarios, and use limited information to create a final project.
3. The pupil is able to engage in discussions about the programming solutions, and use the information gathered to produce a final project.
4. The pupil is able to engage extensively in discussion about the programming solutions, and use the information gathered to produce a final project which includes the use of a range of media.

### Suggestion

[These rubrics are provided as guidelines, and can be adapted to suit your own teaching/feedback methods.](#)



# Observation rubrics grid

Class:		Project					
Pupil name							
		Explore	Create	Share	Explore	Create	Share
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							



(1. Emerging, 2. Developing, 3. Proficient, 4. Accomplished).



# Pupil-led assessment

## Documentation pages

Each project will ask the pupil to create a document to summarise their work.

To have a complete report, it is essential that pupils:

- Document with various types of media
- Document every step of the process
- Take the time to organise and complete their document

It is most likely that pupils can improve on the first document they produce:

- Allow them time and feedback to see where and how they can improve it.
- Ask your pupils to share their documents with each other. By communicating their programming solutions and scientific findings, pupils will develop a deeper understanding of the concepts.

## Self-assessment statements

After each project, pupils can reflect on the work they have done. Use the following page to encourage reflection and set goals for the next project.



# Pupil self-assessment rubric

Name: \_\_\_\_\_

Class: \_\_\_\_\_

Project: \_\_\_\_\_

	Explore	Create	Share
	I documented and used my best reasoning in connection with the question or problem.	I did my best work to solve the problem or question by building and programming my model and making changes when needed.	I documented important ideas and evidence throughout my project and gave my very best when presenting to others.
<b>1</b>			
<b>2</b>			
<b>3</b>			
<b>4</b>			

### Project reflection

One thing I did really well was:

---

One thing I want to improve upon for next time is:

---

# Experimentation Activity

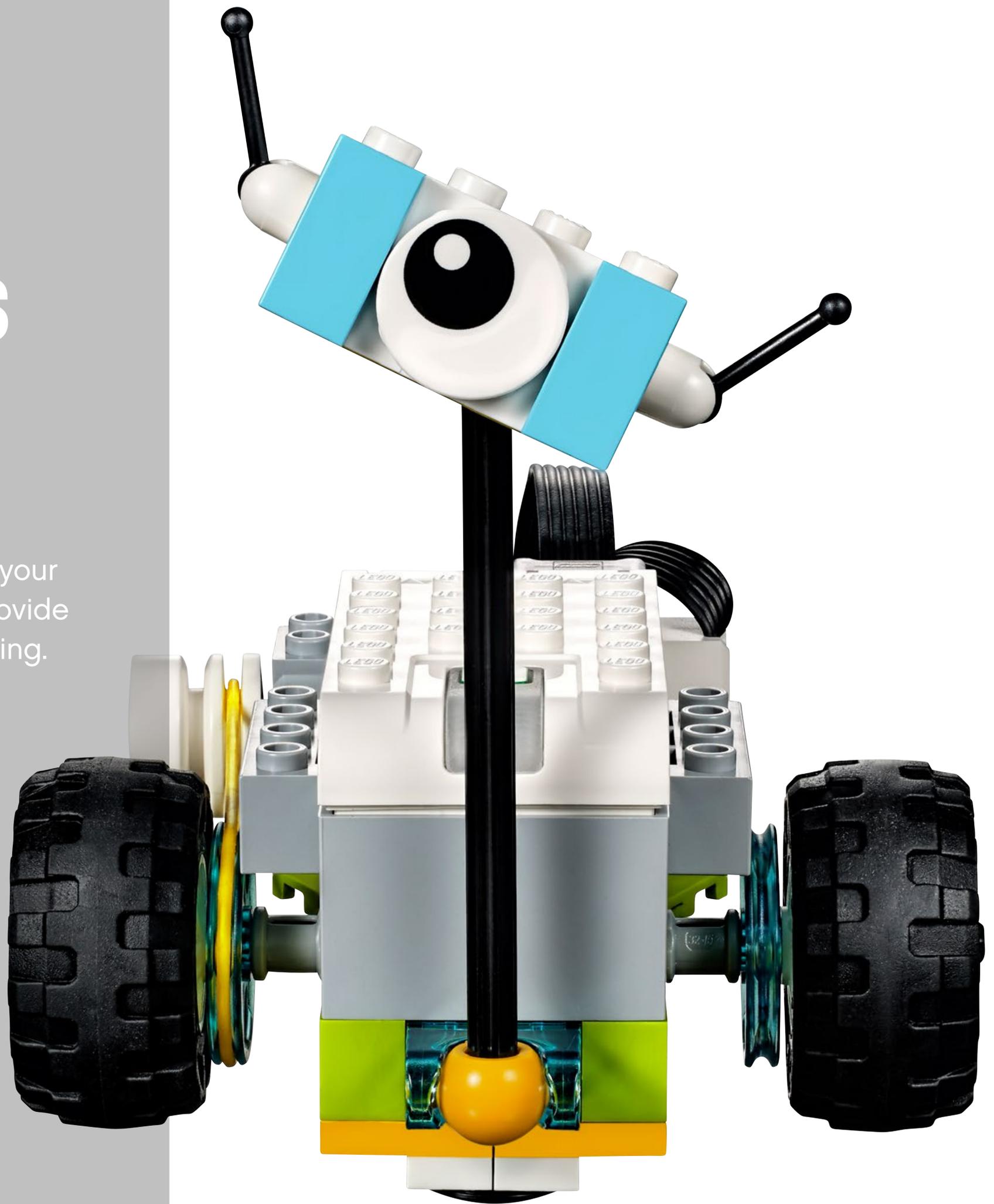
Rethinking Milo's head  
29-35



Experimentation Activity

# Rethinking Milo's head

This activity is designed to quickly introduce your pupils to programming with WeDo 2.0 and provide a playful learning experience around computing.





## Curriculum links and Overview

### Main Learning outcomes

The nature of this activity is to introduce pupils to programming. It is a simple and effective way of encouraging pupils to freely explore programming concepts such as input/output, program strings, programming blocks, and more.

During the activity, many of the requirements of the National Curriculum may be addressed at both Key Stage 1 and Key Stage 2.

### Overview

For this activity, pupils will need to have access to the WeDo 2.0 Core Set, particularly a Smarthub and motor, as they will build a head for Milo.

### Adjustments required

There is no reference to this activity in the Software.  
Building instructions are not provided in the software.  
Building inspiration is provided in this document.

### Keywords

The programming focus is about discovering the ways different elements of WeDo 2.0 interact with each other.

### Milo, the Science Rover



This activity could be carried out before the “Milo the Science Rover” Getting Started project from the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Experimentation Activity

### Preparation (20 min.)

- Read this activity so you have a good idea of what to do.
- Prepare to introduce this activity to your pupils.
- Define your expectations and theirs.
- Determine the end result of this project: Everyone should have a chance to build, program, and document.
- Make sure timing allows for expectations to be met.

### Explore phase (10 min.)

- Start the activity by describing Max and Mia's situation.
- Hold a group discussion.

### Create phase (20 min.)

- Ask the pupils to build a prototype of Milo Science Rover's new head.
- Let them program their model with a simple program.
- Allow the pupils time to make their own experiments and change the parameters of the program.
- Challenge them to discover new programming blocks, and to experiment with them so that they gain an understanding of their functions.

### Share phase (10 min.)

Suggestions for sharing:

- Make sure your pupils take photographs during the process.
- Make sure they write their names and comments in the Documentation tool.
- Ask your pupils to export a document of this activity and to share it with their parents.



## Explore phase

Max and Mia are working in their science laboratory. They are designing prototypes of a head for Milo the Science Rover that will enable it to turn and move around, and they need your help!

To start this activity, you could:

- Show different images of robots or rovers, both from the scientific world or from the fiction world;
- Engage in a discussion with the class on what should be the function of a robot head;
- Show the pupils the video or an image of Milo from the LEGO® Education WeDo 2.0 Software, or present a LEGO model of Milo.

### Suggestion

Introduce them to the [Smarthub](#) and motor from the LEGO Education WeDo 2.0 Core Set. Demonstrate how to attach an axle to the motor and how to create a simple program string to power it.



## Create phase

Ask the pupils to develop solutions for this task:

### 1. Build and program a head that can rotate.

Give the pupils some time to explore the LEGO® elements in the WeDo 2.0 Core Set. Let them work in pairs and experiment with various builds and program strings, or algorithms.

#### ▶ Important

The pupils will experiment with the programming blocks (particularly the green, 'action' blocks). During this process they will discover, through experimentation, the function of each block:

- Ask them to make the motor speed-up and slowdown;
- Ask them to stop the motor after a given time;
- Ask them to rotate Milo's head one way, and then the other;
- Ask them to, if appropriate, explore the use of sensors.
- Ask them to change the color of the LED on the Smarthub.

## Share phase

Bring the pupils together and ask them what they have discovered, and if they have any questions or queries.

- Ask the pupils to select two different builds and programming solutions that will move the head in different ways.
- Ask them to take screenshots of their program strings and photographs of their builds, and to put this information into the documentation tool.
- Ask one or two pairs to show and explain their solutions to the rest of the class.





## Suggested Programming Solutions

### Suggested programming solutions for task 1

In this program, linear programming is used.

This program will turn the motor on in one direction. The motion is unlimited as there is not a Motor Off block to stop it.



In this program, linear programming is used.

This program will turn the motor on in one direction. The motion will last for five seconds and then stop.

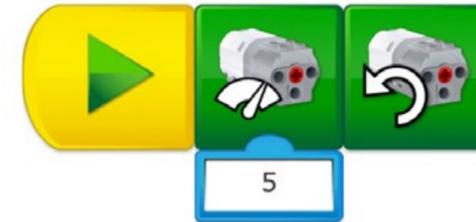




## Suggested Programming Solutions

In this program, linear programming is used.

This program will set the motor power to five (out of 10) and turn the motor on in one direction. The motion is unlimited as there is not a Motor Off block to stop it.

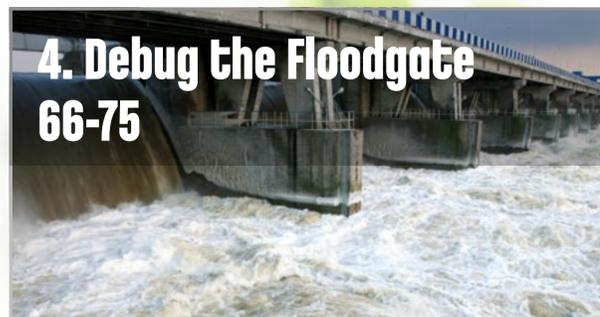


In this program, linear programming is used.

This program will turn the motor in one direction for five seconds and then in the opposite direction for three seconds. It will then stop.



# Guided Projects Extensions overview



## Project Extension 1

# Speed Control

This project is about designing programs that will help a race car driver to determine and control the speed of a race car.





## Curriculum links and Overview

### Main Learning outcomes

During this project, pupils will:

- Work with various forms of input
- Work with various forms of output
- Work with variables
- Control or simulate physical systems

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Use repetition in programs
- Solve problems by decomposing them into smaller parts
- Use selection in programs
- Use logical reasoning to explain how some simple algorithms work
- Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

To make this project feasible, pupils will have to build a joystick to put on their race car to act as a gearbox. To build this, they can use the tilt base instructions in the Design Library.

### Adjustments required

Building instructions for the model used in this project are provided in the software. Building inspiration for the joystick is provided in the Design Library of the software.

### Keywords

The programming focus is on how changes in input, such as changing the state of the tilt sensor, affects output, but other concepts will be covered such as the use of loops and parallel programming.

- Input/Output
- Sensors
- Loop
- Parallel Programming



This project draws on the Speed project of the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is designed to follow the Speed science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the Speed project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How the design of their joystick helped them solve the problem.
- How the program they created helped them solve the problem.
- What elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped explain the process and results of the team.



## Explore phase

Now that Max and Mia have studied the factors that can influence the speed of a car, the manufacturers of a race car are requesting their help to solve a very precise problem.

The manufacturers want to develop a new method for drivers to control the direction and speed of their cars. They want their cars to drive forwards and backwards, and stop. Max and Mia are looking for your help.

Engage in a discussion with the class and allow them to discuss how cars work. Ask them to think of how a driver controls speed using a gearbox transmission in a real car. Allow them to reflect if using programming could be a way to solve the problem.



## Create phase

Ask the pupils to build the model of the Race Car from the provided instructions. Then, let them design a joystick to put on their cars. You can show the pupils where the Tilt base model is in the WeDo 2.0 Design Library for inspiration. Alternatively you can ask them to build one of their own design.

Ask the pupils to develop solutions to one or all of these tasks:

### 1. Develop programs that can make the car travel forwards, backwards and stop, using a joystick.

To have succes with this task, pupils should:

- be reminded how to program the Tilt sensor using the Wait For Block;
- understand how to configure the different states of the sensor;
- use the Repeat Block to have their actions executed more then one time;
- use parallel programming to use different stages of the Tilt sensor simultaneously.

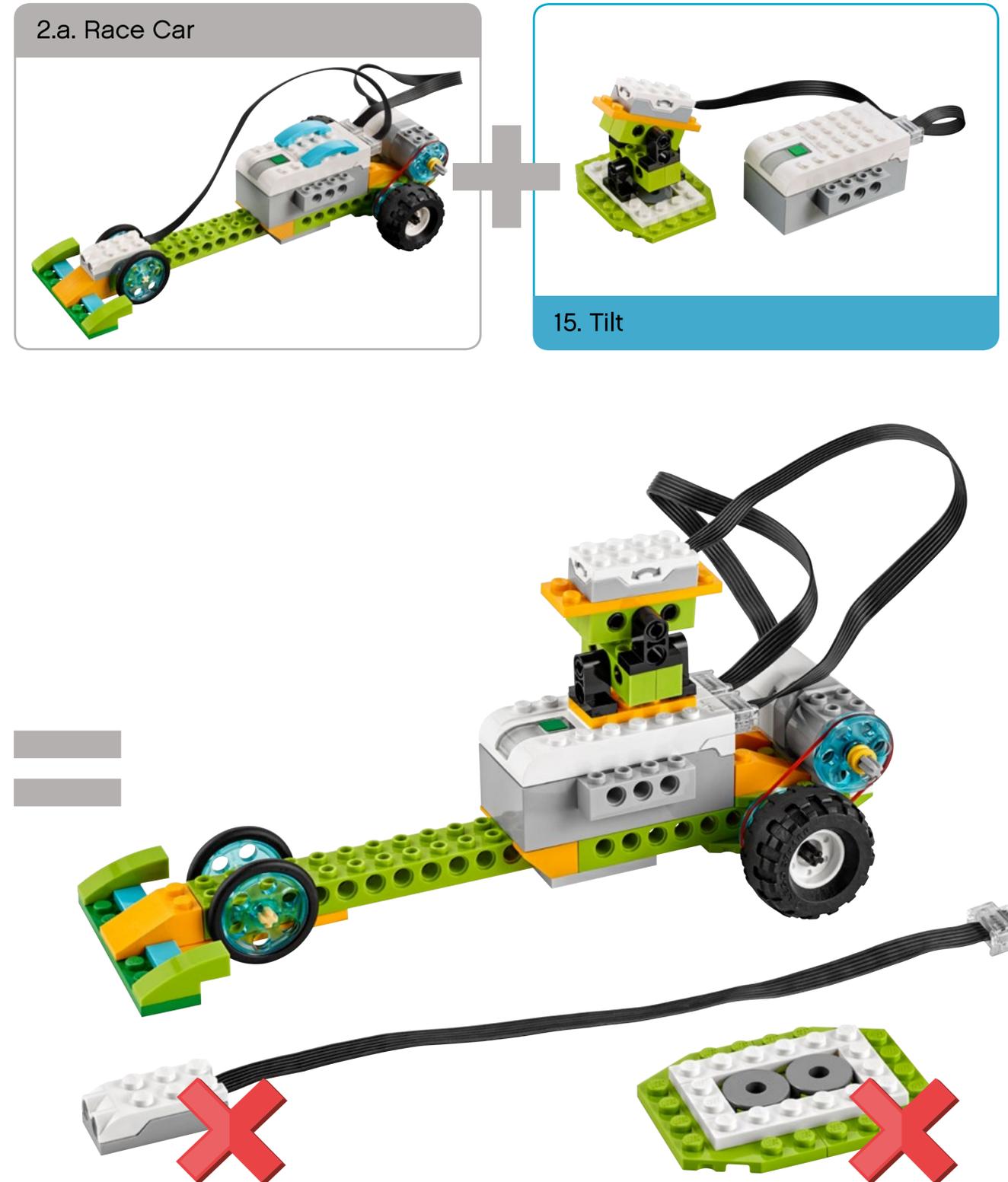
Let your pupils explore different programming solutions to make their car travel forwards and backwards. Remind them that their joystick will need a “neutral” setting for the car to stop!

### 2. Create a program that will make the car drive forwards at two different speeds.

### 3. Create a program that will make the car drive forwards a three different speeds.

#### Important

The pupils may change their builds or their programs to solve the tasks. In this case, they will have to take the Motion sensor off the car so that the Tilt sensor can be used for the joystick.





## Create phase

Here are some suggested programs or guidelines you can use to have a discussion with your pupils.

### Suggested programming solution for task 1

In this program, parallel programming is used.

All program strings that have the Start on A Block will start when just one of them is tapped.

- If the Tilt Sensor is pointing up, the motor will rotate in one direction.
- If the Tilt Sensor is pointing down, the motor will rotate in the opposite direction.
- If the Tilt Sensor is horizontal, the motor will stop.





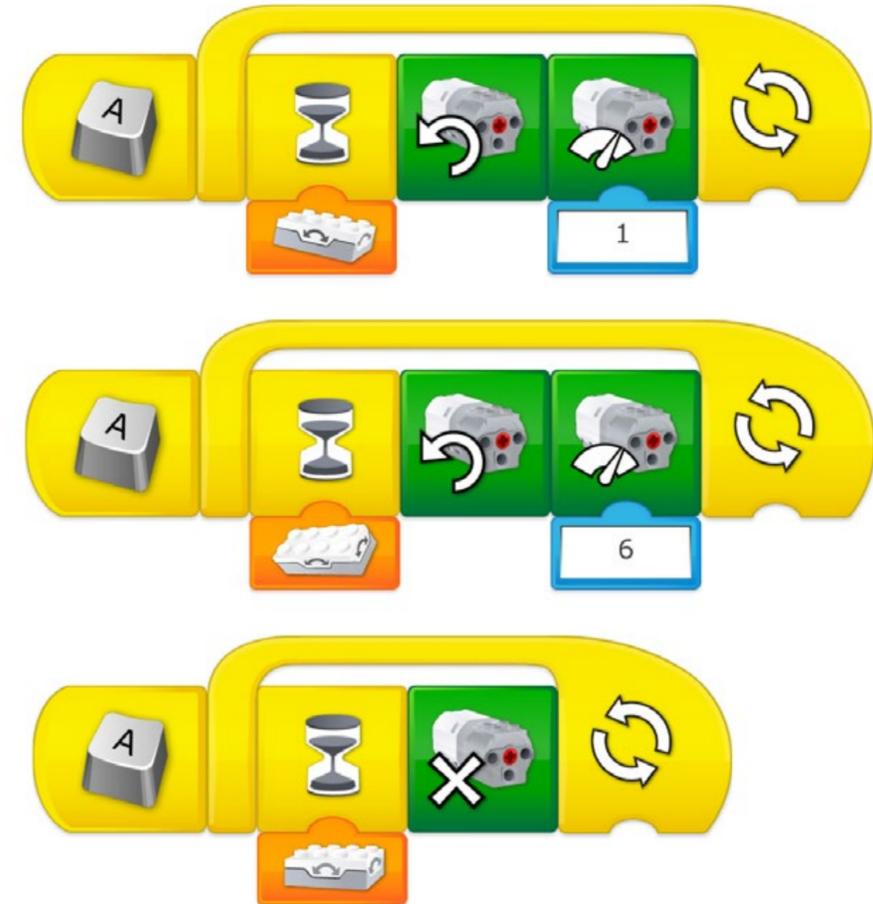
## Create phase

### Suggested programming solution for task 2

In this program, parallel programming is used.

All program strings that have the Start on A Block will start when just one of them is tapped.

- If the Tilt Sensor is pointing down, the motor will rotate one way at speed 1.
- If the Tilt Sensor is on one side, the motor will rotate one way at speed 6.
- If the Tilt Sensor is horizontal, the motor will stop.





## Create phase

### Suggested programming solution for task 3

In this program, parallel programming is used.

All program strings that have the Start on A Block will start when just one of them is tapped.

- If the Tilt Sensor is pointing down, the motor will rotate one way at speed 1.
- If the Tilt Sensor is tilted on one side, the motor will rotate one way at speed 6.
- If the Tilt Sensor is tilted on the other side, the motor will rotate one way at speed 10.
- If the Tilt Sensor is pointing up, the motor will rotate the other way at speed 1.
- If the Tilt Sensor is horizontal, the motor will stop.





## Create phase

### Design further solution

The manufacturers and the drivers of the car loved your ideas, but they would like you to go further and introduce more features.

1. Design a program to use the range of speed of the motor (between 1 and 10) so that the driver has more control over how fast the car goes.
2. Design a visual indicator that tells the driver which speed the car is in.

### Important

The key to this programming problem is to use the Display as an indicator so we know how fast the motor is going. The number on the display acts as an input to the set the motor power (the higher the number, the faster the motor). You may need to show the pupils how to change the state of the Add to Display Block so that it changes to the Subtract from Display Block, simply by pressing on the Block.

### Suggestion

A classroom management tip here is to get the pupils to turn their cars over so they can see the motor change speed before they attempt to control their cars on the table or floor.

### Suggested programming solution

In this program, parallel programming is used.

Two program strings start when the Start on A Block is tapped.

- If the Tilt Sensor is pointing up, the motor will turn in one direction with the motor power set by the number on the display.
- If the Tilt Sensor is pointing down, the motor will turn in the opposite direction with the motor power set by the number on the display.

When one of the Start blocks is tapped, the number on the display is changed to a greater or smaller number, adjusting the power of the motor.





## Share phase

### Complete the document

Make sure your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

### Present results

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

### ► Suggestion

One way of sharing could be to combine two teams and ask them to describe their program strings to each other. Then, ask them to swap their programs so they can try the other team's program on their own model. Ask them to describe the difference in the behaviour.

A close-up photograph of a green tree frog with distinctive red and white markings on its belly, perched on a green stem. The frog's large, dark eyes are prominent, and its skin has a textured, bumpy appearance. The background is a soft, out-of-focus green, suggesting a natural habitat.

**Project Extension 2**

# **Fearless Frog**

This project is about modelling a frog that can recognise and escape from dangerous predators.



## Curriculum links and Overview

### Main Learning Outcomes

During this project, pupils will:

- Work with various forms of input
- Work with various forms of output
- Use repetition in programs

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Use selection in programs
- Use logical reasoning to explain how some simple algorithms work
- Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

Pupils will be asked to replicate as best as possible the behaviours of a frog escaping from a predator. Rebuilding the frogs to better illustrate the animal movement is possible but not required.

### Adjustments required

Building inspiration for the model used in this project is provided in the Design Library of the software.

Building inspiration for adjustment of the model is provided in this document.

### Keywords

The programming focus is on input and output, leading to using two different inputs, but other concepts will be covered.

- Input/Output
- Sensors



This project draws on both the Frog's Metamorphosis guided project and the Predator and Prey open project.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is designed to follow the Frog's Metamorphosis science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the Frog's Metamorphosis project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How did the team use the model of their frog to demonstrate behaviours?
- How the program they created helped them demonstrate behaviours.
- What elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped to explain the process and the results.



## Explore phase

Now that the tadpole has grown into a frog, Max and Mia realise it needs to protect itself from predators. They would like to know more about animals that might prey on frogs. Max and Mia have also discovered that frogs facing a predator have multiple ways to get away quickly to safety!

Engage in a discussion with the class to define what the main predators of frogs are. Discuss with the pupils how the frog might show when it is in danger and when it is safe. Which colours are associated with safety and danger, for example?

Allow the pupils to discuss ways of programming so that their frogs “hop” away when the program is executed.

### Suggestion

You may wish to provide the pupils with a plan of a garden, showing safe areas, and ones where a frog might be in danger. These plans could be quite large, allowing the pupils' models to be placed on them when the programs are executed. Alternatively, the pupils could create their own garden plans.



## Create phase

Ask the pupils to build model of a frog from the provided instructions.

Ask the pupils to develop solutions to one or all of these tasks:

**1. Create a program that will make your frog hop in various ways.**

Ask your pupils to make sure the frog can move at various speeds to escape.

**2. Model what the frog does to show when it is in danger and when it is safe.**

Let the pupils explore different programming solutions to make the frog leap and to show when it is safe, and when it is in danger. The pupils will need to explore different programming solutions to complete this task.

**3. Model how the frog reacts to danger; when it detects a predator, or when it hears a sound.**

Frogs are also in danger at night! As well as using their eyes, they need to listen for danger. Ask the pupils to use sensors to detect the presence of predators. Ask them to create a parallel program and utilise both the Sound Sensor and the Motion Sensor.

**▶ Suggestion**

Ask the pupils to work in silence and in the dark for maximum effect!

**▶ Important**

Pupils will need to connect the Motion sensor to their frogs, and could experiment where the best position is for the placement. You may need to remind the pupils how to program the Motion Sensor using the Wait For Block. They may also use the Repeat Block so that their frog repeatedly moves away from danger.



## Create phase

### Suggested programming solution for task 1

In this program, we are using linear programming.  
The motor speed will be set to 8, and the motor will run for three seconds before stopping.  
This would represent the “normal walking” mode.

### ▶ Important

If the frog needs to reach certain areas of the garden in order to escape danger, pupils could calculate or experiment with timings and direction.



In this program, we are using linear programming.  
The motor speed will be set to 10, and the motor will run for three seconds before stopping.  
This would represent the “escape from predator” mode.





## Create phase

### Suggested programming solution for task 2

In this program, we are using linear programming.

The Smarthub will illuminate red (No. 9), indicating danger. The motor speed will then set to 8 before turning on for three seconds. The Smarthub will then illuminate green (No. 5) and stay illuminated for three seconds.

### ▶ Important

Without the Wait For 3 sec. Block at the end of this program, the green light would only appear for a moment, as it is placed at the end of the program.



In this program, we are using linear programming.

The Smarthub will illuminate red (No. 9), indicating danger. The motor speed will then set to 8 before turning on for three seconds. The Smarthub will then illuminate green (No. 5) and stay illuminated for three seconds. This time we have added a visual indicator using the display.





## Create phase

### Suggested programming solution for task 3

- In this program, we are using parallel programming.
- When any of the Start on Key Blocks are pressed, the two program strings will be executed:
- The first program string (and the frog) will wait for the Motion Sensor to detect movement in front of or behind it (depending on where the Motion Sensor is positioned). When a movement is detected, the motor speed will be set to 10, and the motor will run for three seconds before stopping. This sequence will then be repeated.
- The second program string (and the frog) will wait for the sensor on the device to detect a sound. When a sound is detected, the motor speed will be set to 10 and the motor will run for three seconds before stopping. This sequence will then be repeated.



### ► Important

The Sound Sensor is not located on the Smarthub. It is in fact the microphone of your device. You may need to show the pupils how to use the Sound Sensor with the Wait For Block.



## Create phase

### Use the model further

Predators don't give up on their prey so easily. Sometimes animals will have to use other strategies than simply running to save their lives! Discuss the different ways that frogs might escape from predators. Identify the different types of movements a frog can make, such as walking, leaping, swimming, climbing, etc.

#### 1. Model the reaction of the frog so that it moves when a predator comes near – using a different motion than running or leaping away.

For this task, pupils will probably need to rebuild their frog to model different movements. Ask them to look at the base models in the Design Library for inspiration.

Ask them to use the Motion Sensor, so they can integrate the functions that they have learned about in this project.

#### 2. Model the behaviour of a frog predator.

Ask a group of pupils to build a predator, for example a snake, and to program its behaviour.

### Suggestion

You could pair up the teams and ask them to discuss the relationship between the predator and the prey.



## Share phase

### Complete the document

Make sure that your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

### Present results

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

### Suggestion

Ask the teams to discuss the environment where both predator and prey live. Encourage them to focus on the relationship and the life cycle of both animals.

## Project Extension 3

# Dancing Bees

This project is about modelling the dance of a bee that communicates the distance and location of pollen-bearing flowers, and how to return to the hive after foraging.





## Curriculum links and Overview

### Main Learning Outcomes

During this project, pupils will:

- Use sequence in programs
- Use repetition in programs
- Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Work with various forms of input
- Work with various forms of output
- Use selection in programs
- Use logical reasoning to explain how some simple algorithms work
- Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

To make this project feasible, pupils will need to model the behaviour of a bee as it flies from a flower to the hive. Pupils could also be asked to describe the limitations of what they can model.

### Adjustments required

Building instructions for the model used in this project are provided in the software.

### Keywords

The programming focus is on sequence and loop, but other concepts will be covered.

- Sequence
- Loop
- Output



This project draws on the Plants and Pollinators project of the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is designed to follow the Plants and Pollinators science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the Plants and Pollinators project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How did the team use the Flower model to demonstrate bee behaviours?
- How the program they created helped them to demonstrate behaviours.
- What elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped to explain the process and their results.



## Explore phase

Did you know that when bees are on their pollen-gathering mission, they dance? Max and Mia are researching how and why they do this. They need your help to find out more information about this.

Allow time for pupils to reasearch and document the behaviour of bees.

Then, engage in a discussion with the class and allow them to discuss ways of programming so that their bees “dance” during their flight to collect the pollen.

### Suggestion

You could show the pupils some video footage of bees in flight and of them gathering pollen. Alternatively, the pupils could use their devices to search for, and view, a range of videos. It's a good idea for you to research some videos first, so you can provide the pupils with a list of key search words. These could include:

- dancing bees
- do bees dance
- flying dance of bees
- bees collecting pollen



## Create phase

Let the pupils build the model of the pollination from the provided building instructions.

Ask the pupils to develop solutions for one or all of these tasks:

**1. Create a program that simulates a bee dancing on its journey to a flower or on its return journey to the hive after collecting pollen.**

Pupils will need to explore different programming solutions to complete the task. Encourage them to use the Repeat Block and identify how this makes a difference to their programs. There will be more than one solution to the task.

**2. Adapt your program so it indicates to the beekeeper when a bee is in the hive, and when it has reached a flower.**

Pupils will have to define where the hive is situated on the path of their bee. To find solutions to this task, encourage them to think of using an audio alert, a visual signal, or a colour signal. For example, different colours could indicate whether the bee is at the flower gathering pollen, or back at the hive.

### ▶ Suggestion

You may need to remind the pupils how to program the Motion Sensor using the Wait For Block, and show them how to configure the different states of the sensor.

### ▶ Important

The pupils should be allowed to change their builds or their programs to solve these tasks.





## Create phase

### Suggested programming solutions for task 1

In this program, we are using linear programming.

- The power of the motor is set to 4, and the bee is flying in one direction.
- When the Motion Sensor detects movement in front of it, the bee will stop and buzz.
- The bee will then dance, moving in one direction and then in the other direction.
- The motor will then stop.



In this program, we are using sequencing.

The speed is set, and the bee is flying. When the Motion Sensor detects movement the program will stop the motor, play a sound, and send a message "D" to the next line. The programming string starting with Start on Message Block with input "D" will be activated: the bee will dance back and forth twice before stopping.



### Important

This second program does exactly the same as the previous one, except the bee will move back and forth twice. The advantage of using a loop in this situation is that we can repeat a sequence without having to add new blocks to the program. In that sense, we can say that this program is more efficient than the previous one.



## Create phase

### Suggested programming solution for task 2

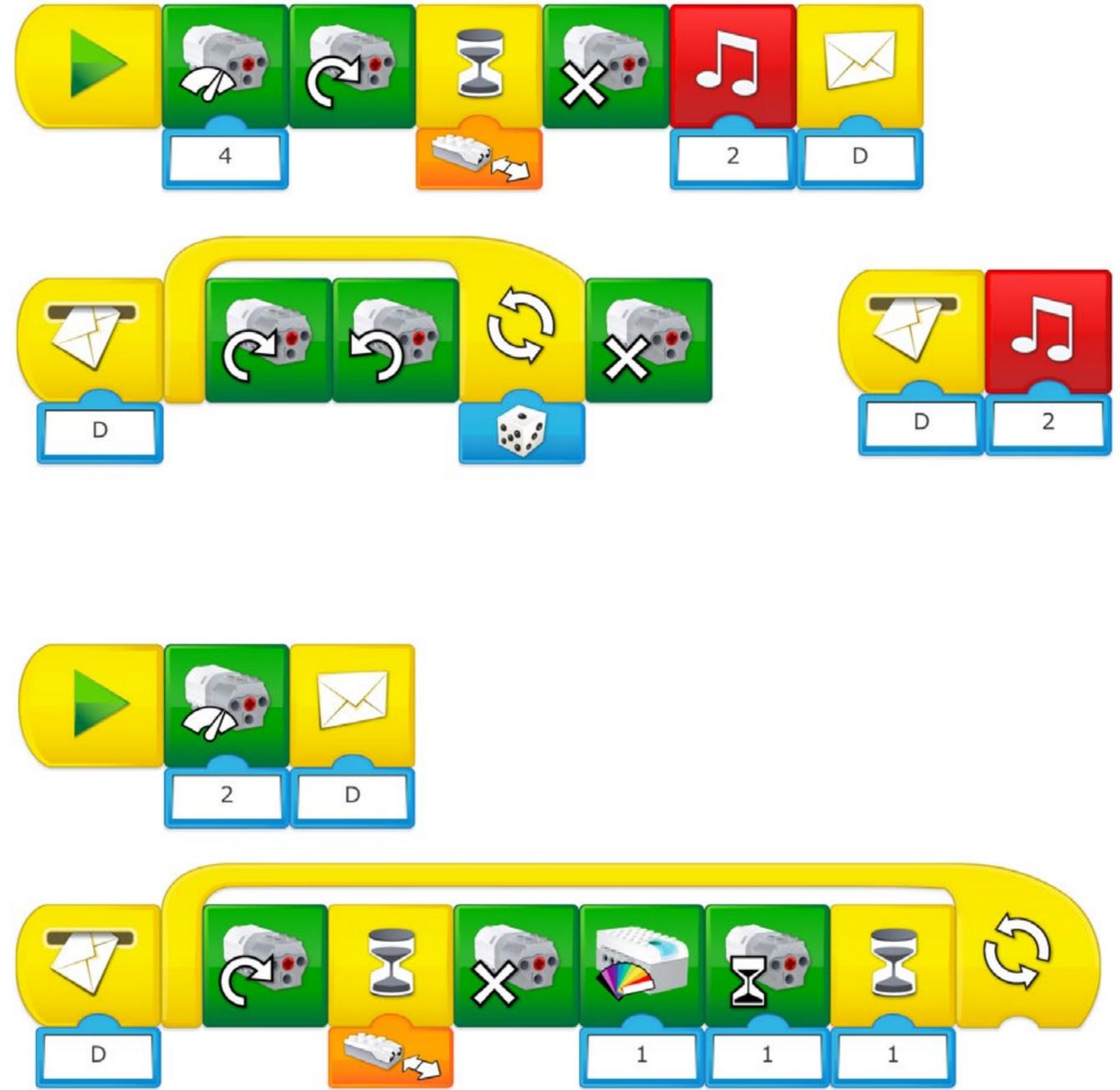
In this program, we are using sequencing.  
 When the Start Block is pressed, the motor power is set to 4 in one direction and the bee will fly towards the flower. When the Motion Sensor detects the bee, the motor will stop and the bee will make a buzzing sound.

The message "D" will be sent, triggering two elements at the same time:

- The motor will then start again, causing the bee to move back and forth for a random number of times (between 1 and 10). The bee will then stop.
- Once again the buzzing sound will play.

In this program, we are using sequencing.  
 When the Start Block is pressed, the power level will be set to 2, and a message "D" will be sent to the next line of the program.  
 When "D" is received, the bee will begin to fly. When the Motion Sensor detects an object (the bee), the motor will stop and the Smarthub will light up Pink (No. 1). The motor will then start again for one second, and stop at a distance from the flower, this should be the position of the hive.

The process is then repeated.





## Create phase

### Use the model further

You might have discovered that the bees are dancing to communicate important information to other bees. For example, the length of time they waggle communicates how far away the pollen is. A one-second waggle could mean “Fly for 750 meters”. When bees waggle, they also angle toward the pollen source, using their heads to point the way. Honeybees also dance in the darkness of their enclosed hives. So hive-mates won’t see this dance. Rather, they’ll hear the vibrating sound a dancing bee makes. They will even touch the dancer with their antennae to get a better sense of vibrations and angle.

Engage in a discussion with the class. Ask them to describe the movements that they have incorporated into the bee dance.

### 1. Create your own coding system linked to the bee movement and communicate the position of the flower to the other team.

For example, pupils could decide that:

- A single clockwise rotation of the bee is equivalent to one metre forwards
- A single anticlockwise rotation of the bee is equivalent to one metre backwards
- A bee sound for turning 90° left
- A second sound for turning 90° right
- etc.

Ask the teams to create their own key for their code. When you feel your teams are ready, Ask two teams to work together so that one team can decode the dance of the other.



## Share phase

### Complete the document

Make sure that your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

### Present results

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

### Suggestion

Just as bees work together in large groups, it makes sense here to engage pupils in a very dynamic process of sharing and interacting with other teams. Encourage them to compare their programs, adjust their strategies, and share their ideas to enrich the experience!

A large concrete floodgate structure with turbulent water flowing through it. The structure consists of several vertical concrete pillars supporting a horizontal gate. The water is churning and white with foam as it passes through the narrow openings. The sky is clear and blue.

**Project Extension 4**

# **Debug the Floodgate**

This project is about designing programs to correct errors in the way a floodgate is working.



## Curriculum links and Overview

### Main Learning Outcomes

During this project, pupils will:

- Debug programs that accomplish specific goals
- Use logical reasoning to detect and correct errors in algorithms
- Control or simulate physical systems

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Use sequence in programs
- Work with various forms of input
- Work with various forms of output
- Use repetition in programs
- Solve problems by decomposing them into smaller parts
- Use selection in programs
- Use logical reasoning to explain how some simple algorithms work
- Create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

To make this project feasible, you will have to show “incorrect” programs to your pupils and ask them to find solutions so the models work as expected.

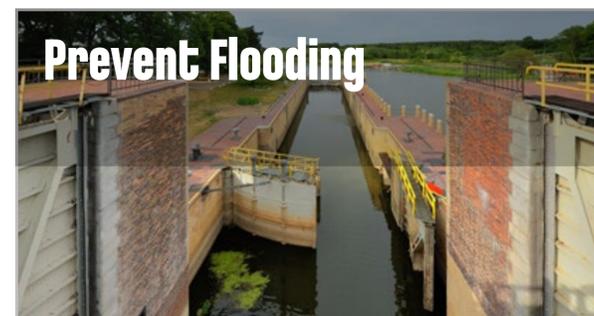
### Adjustments required

Building instructions for the model used in this project are provided in the software. Building inspiration for adjustments to the model is provided in this document.

### Keywords

The programming focus is initially on debugging, but other concepts will be covered as the project develops, such as Wait for, sequence and loop.

- debugging



This project expands on the Prevent Flooding project in the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is designed to follow the Prevent Flooding science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the Prevent Flooding project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How the design of their floodgate helped them solve the problem.
- How were they able to identify the problems in the programs provided.
- Which elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped to explain the process and the results.



## Explore phase

Max and Mia are happy that the floodgate on the river operates automatically. Suddenly, they realise there has been a malfunction and that there is a problem in the floodgate control room! The gate is opening and closing at the wrong times, and is not responding properly to changes in the weather. This is causing the river water to flow the wrong way.

The gate needs to be open when it is raining, and closed when the weather is fine. It needs to do this smoothly and safely, and allow time for the river to reach a safe level.

Engage in a discussion with the class and discuss how program bugs can occur. Discuss the possible consequences of such a malfunction in systems or machines in general.



## Create phase

Let the pupils build the model of the floodgate from the provided instructions.

Ask the pupils to develop solutions for one or all of these tasks:

### 1. Debug the first program of the floodgate.

When operating correctly, the floodgate should open when the weather is fine and close when it is raining.

- Show the pupils the program that needs to be debugged.
- Ask them to use the Bubble to annotate the problem the flood gate has.
- Discuss the various ways to reprogram the floodgate so that it works properly.
- Ask the pupils to correct the program to match the expected behaviour.
- Ask the pupils to document their process.

### ► Suggestion

This project could be completed in small or larger groups.

### 2. Debug the second program of the floodgate.

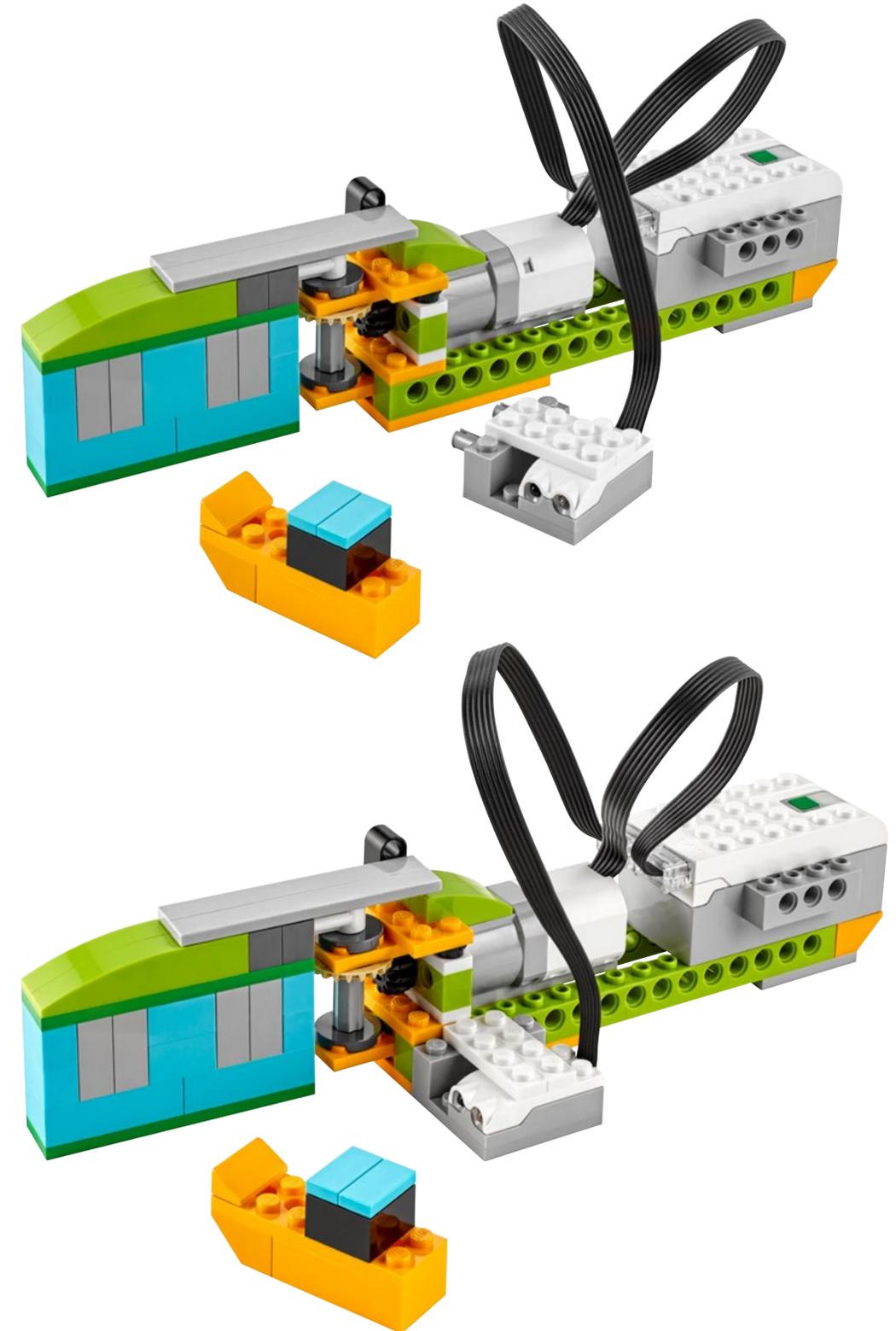
Some floodgates are able to open and close to allow river traffic through. When operating correctly, the floodgate should recognise when a boat is present and allow it through the gate.

### 3. Debug the third program of the floodgate.

If operating correctly, when a boat has passed through the gate, the gate should indicate a countdown from five to zero, and then close by itself.

### ► Important

The pupils could change their builds to solve the tasks, but the tasks have been designed so that only the programs need to be changed.





## Create phase

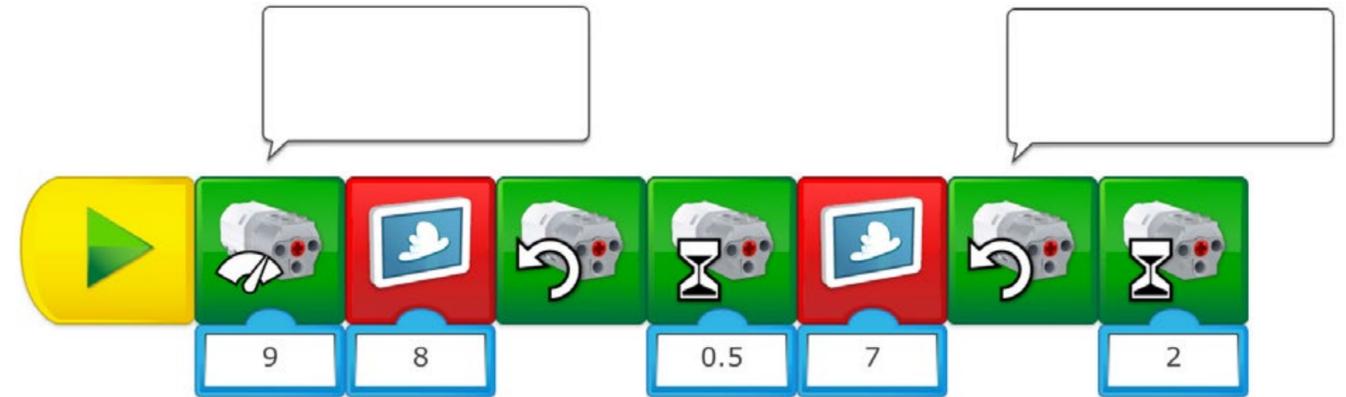
### Suggested program to debug for task 1

The first program is the one that contains elements to be corrected in order for the floodgate to operate correctly. The elements that could be changed are:

- The power of the motor is very high, which might not be easy to control.
- The motors are turning in the same direction so the floodgate can open, but it can not close.

### ▶ Suggestion

Ask your pupils to try this program with their model so they can see what the issues are. Ask them to use the Comment Bubbles to write what they think the problem is.



### Suggested programming solution for task 1

In this program, we are using linear programming.

The motor power is set at 1 (out of 10). When nice weather is displayed (Image No. 7) the floodgate should open for three seconds. When the bad weather is displayed (Image No. 8) the floodgate should close for three seconds.



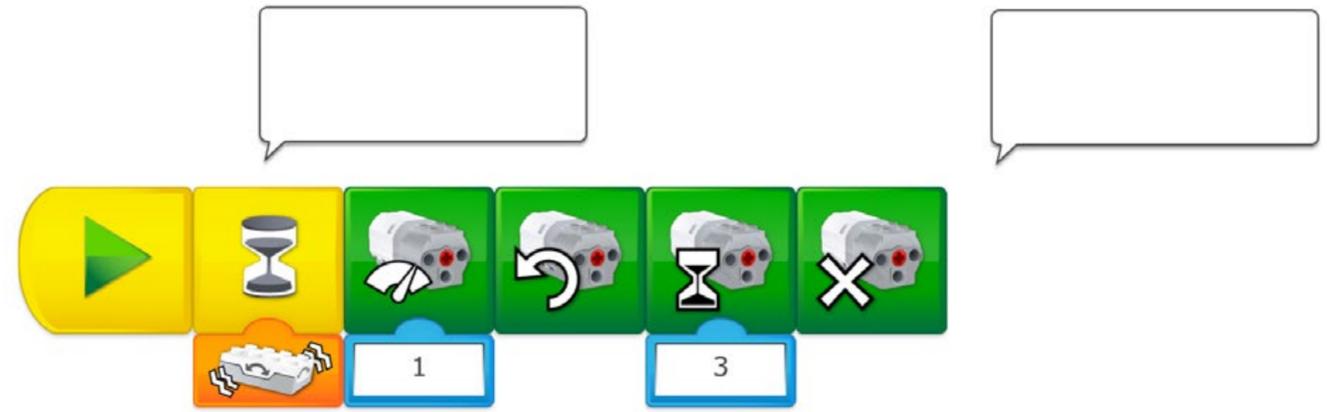


## Create phase

### Suggested program to debug for task 2

The first program is the one that contains elements to be corrected in order for the floodgate to operate correctly. The elements that could be changed are:

- The floodgate has a Motion Sensor to detect the presence of boats, using a Wait for Any Tilt Block to start the programming string will not work.
- The floodgate does not have a sequence to close itself.



### Suggested programming solution for task 2

In this program, we are using sequencing.

When an object (a boat) passes in front of the Motion Sensor, the motor power is set to 1 (out of 10). The floodgate should open for three seconds.

The message “abc” is sent to start the second program string that will run the motor in the opposite direction for three seconds before stopping, closing the gate.



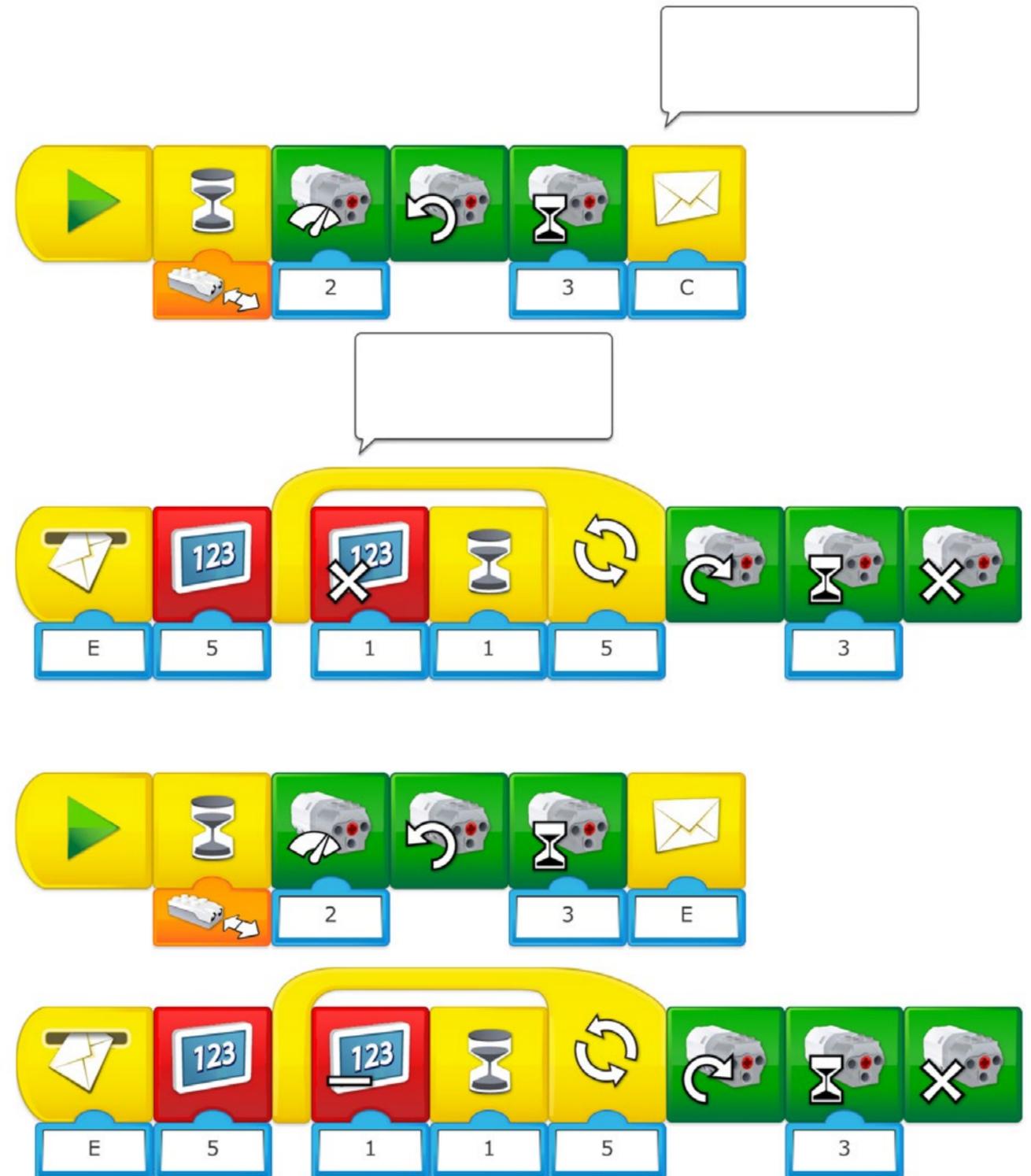


## Create phase

### Suggested program to debug for task 3

The first program is the one that contains elements to be corrected in order for the floodgate to operate correctly. The elements that could be changed are:

- The message “C” is never received by another sequence
- In the loop, we multiply the number 5 on the display by one, therefore the display always shows 5 and not 4, 3, 2, 1, 0 as it should do.



### Suggested programming solution for task 3

In this program, we are using sequencing.

When an object (a boat) passes in front of the motion sensor, the motor power is set at 2 (out of 10). The floodgate should open for three seconds. The message “E” is sent and received by the next program string. The number 5 appears on the display. Then, each time the loop is executed, the number counts down by one to create the count down 4, 3, 2, 1, 0. At that point, the motor will be activated in the opposite direction for for three seconds, and then stop.



## Create phase

### Design further solutions

As in all control rooms, employees sometimes have to be replaced for various reasons. When that happens, the new person needs to learn how the machinery works, or how it should be working!

#### 1. Create a program to be debugged by another team.

Every team should write a description of the expected behaviour of the floodgate, specifying all the parameters such as the position of the sensor, the gate etc., so that the other team can understand the situation.

Pair the teams up, and ask each team to debug the program the other team has prepared. Ask them to document the process. Videos are a great way to do this.

### ▶ Important

Make sure that the teams do not create programs that are impossible to debug: only two or three parameters should be “wrong”. Have some examples of your own prepared in case some teams are struggling to create their own.



## Share phase

### **Complete the document**

Make sure that your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

### **Present results**

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

## Project Extension 5

# Rescue Count

This project is about designing programs that can help to count the number of rescued animals.





## Curriculum links and Overview

### Main Learning outcomes

During this project, pupils will:

- Work with various forms of input
- Work with various forms of output
- Control or simulate physical systems

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Use sequence in programs
- Use repetition in programs
- Solve problems by decomposing them into smaller parts
- Use selection in programs
- Use logical reasoning to explain how some simple algorithms work
- Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

To make this project possible, pupils may have to build a Motion Sensor on their helicopter to help them count the number of pandas they are rescuing.

### Adjustments required

Building instructions for the model used in this project are provided in the software. Building inspiration for adjustments to the model is provided in this document.

### Keywords

The programming focus is on variables, loops and outputs, but other concepts will be covered, such as the use sequencing and parallel programming.

- Variable
- Loops
- Input/Output



This project expands on the Drop and Rescue project in the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is designed to follow the Drop and Rescue science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the Drop and Rescue project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How the design of their helicopter helped them to solve the problem.
- How the program they created helped them to solve the problem.
- What elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped to explain the process and results.



## Explore phase

Max and Mia have just heard that a natural disaster has occurred. A rescue helicopter has been sent to the scene to save any animals that have survived. They are wondering how the helicopter pilot and the crew will keep track of all the animals they rescue on the mission.

They need your help to create a range of programs that will help the pilot and the crew to complete this task.

Engage in a discussion with the class and allow them to discuss ways in which they could help the helicopter rescue teams.

### **Important**

This project is about extending programming skills to include the use of a sensor and sequencing. It is also about comparing the efficiency of programs. The pupils will create an automated “counting-in” system, but may decide, after some testing, that a manual system would be more effective.



## Create phase

Let the pupils build the model of a helicopter from the provided instructions. Ask them to develop solutions to one or all of these tasks:

**1. Create a program that will assist the crew in counting the animals that are rescued, without using a sensor.**

Pupils should create a range of programs that add to the display of the software each time the helicopter winch reaches the top. They may also create programs which contain other outputs such as light and sound. Pupils may use the same animal to simulate each rescue, or build different animals with the remaining bricks.

**2. Create a program that will assist the crew in counting the animals that are rescued, using a Motion Sensor.**

Ask the pupils to try using a Motion Sensor in order to count the animals that have been rescued. Ask them to decide on the best place to mount the sensor. Encourage them to reflect on their solution and decide if it is better (or not) than the solution that they reached in the previous task.

**3. Adjust your program and sensor to count the number of holes the helicopter can detect in the ground.**

Sometimes, animals will seek refuge in holes in the ground. By repositioning the Motion Sensor to point downwards, pupils can detect holes in the ground represented by, for example, spaces between desks.

**► Important**

The key to this programming problem is to use the Motion Sensor in a different way. It needs to be programmed so that it responds to movement away from it (so it can see holes or hollows).

**► Suggestion**

Recap on how to program the display block and how to add to the display “counter”. Discuss various ways in which numbers could be added to the display (manual and automatic input).





## Create phase

### Suggested programming solution for task 1

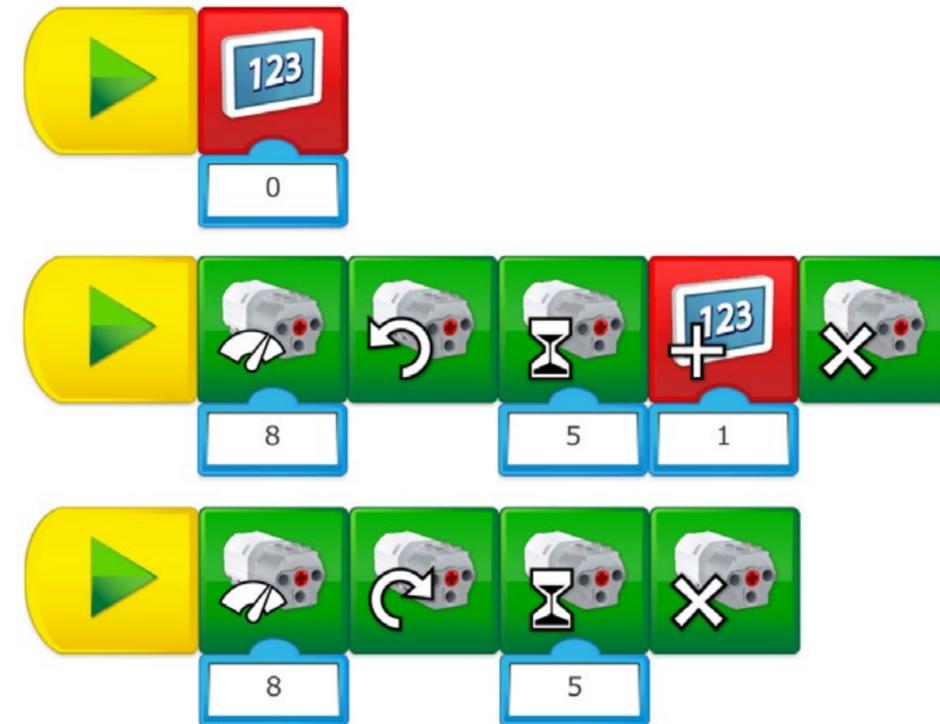
In this program, we are using three program strings.

Each program will run when the appropriate Start Block is pressed. The first program string sets the display to zero. The second program string sets the power of the motor to 8 and turns the motor in one direction for five seconds. This action should bring the winch up. When the winch reaches the top, a count of one is added to the value on the display. The third program string sets the power of the motor to 8 and runs it in the opposite direction for five seconds. This action should send the winch down.

### Important

Because the behaviour of the winch depends on which direction the string was wound onto the axle, it is possible that you will need to invert the motor direction in program strings 2 and 3.

Pupils could also decide to use a fourth string to control counting separately.





## Create phase

### Suggested programming solution for task 2

In this program, we are using four program strings.

Each program will run when the appropriate Start Block is pressed.

The first program sets the display to zero.



The second program string starts the motor in one direction to bring the winch up. The motion sensor, placed on the helicopter, will wait for an object (a panda for example) to come closer to it. When that is detected, a count of one will be added to the display, the green light (No. 5) will then light up on the Smarthub for one second.



The third program string will stop the motor.



The fourth program string runs the motor in the opposite direction. This action should lower the winch.





## Create phase

### Suggested programming solution for task 3

In this program, we are using two program strings.

Each program will run when the appropriate Start Block is pressed.

The first program string will set the display to zero.



The second program string begins with the Wait For Block. In this case, it is waiting for motion away from the sensor. When motion away from the sensor is detected, a count of one will be added to the display, the green light on the Smarthub (No. 5) will then illuminate for one second.



### ► Important

For this program to work:

- The Motion Sensor on the helicopter needs to be pointing downwards.
- The Motion Sensor needs to be pointing at the surface of a desk when the program is started.
- The helicopter (with the sensor) then needs to be moved by hand over a hollow, for example, a space between desks.



## Create phase

### Design further solutions

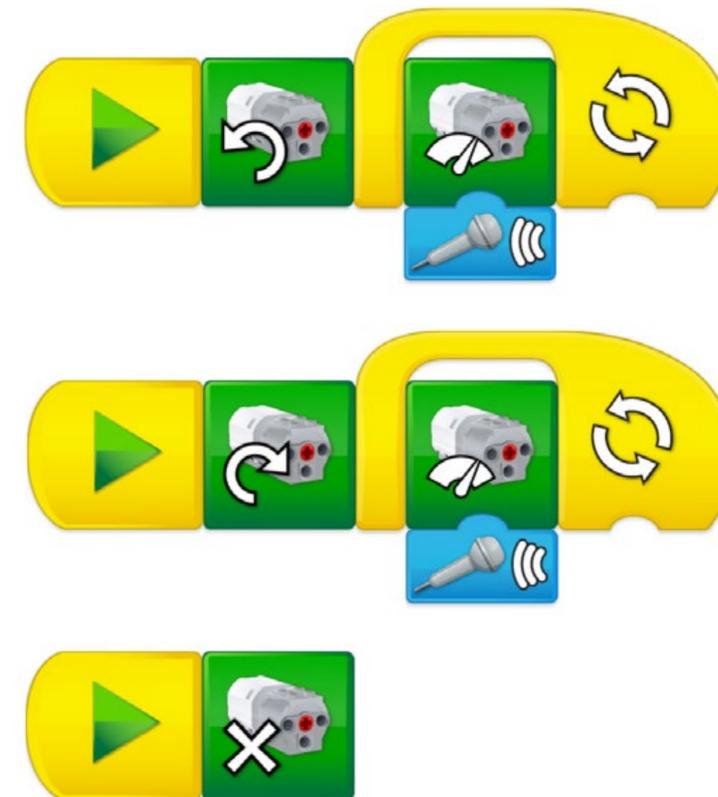
Rescue missions are often hazardous. When the lives of animals and people are at stake, rescue equipment needs to be controlled thoughtfully and accurately. Raising a winch at the right speed is just one of the details to consider.

#### 1. Create a program that can adjust the speed of the winch reel according to the situation: slowly for precision and quickly for emergency purposes.

- Ask the pupils to explore different ways of solving this problem. Various options can be explored:
- Program manual changes in the power of the motor.
- Program motor power to change proportionally according to an input, such as the level of sound. Test these solutions and evaluate which is the most efficient. Try using display input as the variable instead of sound.

#### ▶ Suggestion

Pupils could use the program shown on this page and incorporate it into a larger program that will also count the number of rescued animals. The Power of the motor is controlled by the sound level detected by the microphone. The louder the sound, the more power the motor has.





## Share phase

### **Complete the document**

Make sure that your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

### **Present results**

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

## Project Extension 6

# Reverse and Recycle

This project is about designing programs to make a recycling truck safer for the driver and pedestrians.





## Curriculum links and Overview

### Main Learning outcomes

During this project, pupils will:

- Work with various forms of input
- Work with various forms of output
- Work with variables
- Control or simulate physical systems

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Use sequence in programs
- Debug programs that accomplish specific goals
- Use repetition in programs
- Use selection in programs
- Use logical reasoning to explain how some simple algorithms work
- Use logical reasoning to detect and correct errors in algorithms
- Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

To make this project feasible, pupils will need to build a recycling truck that can move forwards and backwards as well as using a Motion Sensor as an input.

### Adjustments required

Building instructions for the model used in this project are provided in the software. Building inspiration for adjustments of the model is provided in this document.

### Keywords

The programming focus is on output, variables and loop, but other concepts will be covered.

- Input/Output
- Loops
- Variables



This project expands on the Sort to Recycle project in the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is designed to follow the Sort to Recycle science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the Sort to Recycle project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How the design of their truck helped them to solve the problem.
- How the program they created helped them to solve the problem.
- What elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped to explain the process and the results.



## Explore phase

Max and Mia have noticed that the driver of the recycling truck is having problems when reversing, as it is difficult for him to see behind the truck. The main problem he has is knowing when to stop. He needs to know when to brake so that the truck can stop safely, with enough room to tip the load.

Engage in a discussion with the class and allow them to discuss ways in which trucks signal their presence.

### Suggestion

You may want to point out to the pupils that their solutions could include: lights, words on the display, and/or images.



## Create phase

Ask the pupils to build a model of a recycling truck from the provided instructions.

Ask the pupils to develop solutions for one or all of these tasks:

**1. Help the driver by creating a program that will make a sound when the truck approaches an obstacle.**

The key here is to use two separate programs: one for stopping and one to make a sound. Otherwise the linear aspect of the program strings may result in the truck not stopping on time.

**2. Add something to your program so that, as well as a sound, the driver has a visual indication of when to reverse and when to stop.**

The pupils should explore as many solutions as they can, and decide on the best solution to document. There will be more than one solution to the task.

### ► Important

Pupils will have to use a Motion Sensor to develop solutions for this project. Also, it is possible to do the project in two ways:

- Moving the truck slowly towards an obstacle by hand. With this solution, few modifications to the truck are required.
- Modifying the truck to drive itself. This solution requires a longer rebuild and could be used as a differentiation option.





## Create phase

### Suggested programming solution for task 1

In this program, we are using linear programming.

When the Start on Play button is pressed, sound No. 19 will be repeated until the Motion Sensor detects an object in front of it. At that moment, sound No. 8 will be played.

### ▶ Important

For this program to work:

- The Motion Sensor on the truck will need to point at a wall or a vertical surface (obstacle).
- The truck will need to be moved towards the obstacle.





## Create phase

### Suggested programming solution for task 2

In this program we are using linear programming.

When the Start on Play Block is pressed, the green light (No. 5) on the Smarthub will light up. The sound No. 19 will be repeated until the Motion Sensor detects an obstacle. At that moment, the red light (No. 9) will light up for three seconds.

### ▶ Suggestion

Other output such as display text or display images could be used as a solution to warn the driver that it is time to stop.



In this program, we are using linear programming.

When the Start on Play Block is pressed, the green light (No. 5) of the Smarthub will light up. The sound No. 19 will be repeated until the Motion Sensor detects an obstacle. At that moment, the message "abc" will be sent.

Two actions will then take place simultaneously

- The red light (No. 9) will light up for three seconds (the end of the first program string).
- The sound No. 8 will play (the Start on message string).





## Create phase

### Design further solutions

Precision is a skill that drivers have to master. They often have to stop their truck as close as possible to a specific position.

#### 1. Make the truck even safer by automatically slowing it down when it approaches an obstacle.

Pupils will need to explore different programming solutions to complete this task. Once the pupils have explored the concept of mapping speed to distance, they can begin to incorporate this into their programs. Their final solutions may include aspects of project extension 1.

#### ► Important

You will need to show the pupils how to map speed to distance, i.e., the closer an object moves towards the motion sensor, the slower the speed of the motor becomes. It is not possible to measure how far away an obstacle is from the sensor, and therefore not possible to stop the motor when the object is, for example, 3 cm away. Therefore, the pupils will need to calculate the time it takes for their trucks to be at “stopping distance”.

#### ► Suggestion

In the suggested program, when the obstacle is detected by the Motion Sensor, the motor will stop for one second and then run for an extra second at motor power 2 before stopping again. This interval of time can be adjusted so that the truck can stop even closer to the obstacle.





## Share phase

### **Complete the document**

Make sure that your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

### **Present results**

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

# Open Projects Extensions overview



A photograph of a warehouse interior. In the foreground, a red and black forklift is parked, facing right. The background shows long aisles of high industrial shelving units (pallet racks) filled with cardboard boxes. The floor is a light-colored concrete with yellow safety lines. The lighting is bright and even.

Project Extension 7

# Smart Lift

This project is about designing programs to automate a factory warehouse.



## Curriculum links and Overview

### Main Learning Outcomes

During this project, pupils will:

- Use sequence in programs
- Work with various forms of input
- Work with various forms of output
- Control or simulate physical systems
- Use repetition in programs

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Debug programs that accomplish specific goals
- Use selection in programs
- Work with variables
- Use logical reasoning to explain how some simple algorithms work
- Use logical reasoning to detect and correct errors in algorithms
- Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information
- Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

In order for this project to be feasible, as it is an extension of an open project to which there are no specific models associated, it is suggested that pupils build the steering base with forklift modification taken from the Design Library.

### Adjustments required

Building inspiration for the model used in this project is provided in the Design Library of the software.

### Keywords

The programming focus is on input, output, sequencing, and collaboration, but other concepts will be covered.

- Input/Output
- Sequencing
- Collaboration



This project expands on the “Moving Materials” Open Project in the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is designed to follow the Moving Materials science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the Moving Materials project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How the design of their forklift helped them to solve the problem.
- How the program they created helped them to solve the problem.
- What elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped explain the process and the results.



## Explore phase

Max and Mia are impressed by factories and warehouses that use automated robots to carry parcels from one place to another. A factory owner would like to experiment with an automated parcel moving system.

Engage in a discussion with the class. Allow pupils some time to find videos of automated factories. Encourage them to discuss the different ways in which robots operate in warehouses.

### ▶ Suggestion

You could show the pupils some video footage of automated factories and warehouses where robotic vehicles are used to transport items from one part of the factory to another. Alternatively, the pupils could use their devices to search for videos and information online.

It's a good idea for you to research some videos first, so you can provide the pupils with a list of key search words. These could include:

- warehouse robots
- automated warehouse
- robot forklift

### ▶ Important

If pupils have not used the Tilt, Motion, or Sound Sensors before, you may wish to introduce their functions before or during this project.



## Create phase

Ask the pupils to build a model of a forklift. They could build solutions based on the inspirational models in the Design Library or from their own imagination.

Ask the pupils to develop solutions to one or all of these tasks:

**1. Design a program for a forklift so that it can drive to a parcel, stop, collect the parcel and transport it to another part of the factory.**

For this task, pupils will need to make sure that their forklift incorporates the Motion Sensor. They may also need to adapt their parcel, or to build a “stop at” object that the sensor can detect.

### ► Important

Pupils could build a forklift that is able to turn when driven by a motor, or one which needs turning by hand. The second option would simplify the task.

**2. Design a program that enables the forklift to drive forwards when the forks are lowered and backwards when they are raised.**

It must also give visible signals so that the factory workers know when to load and unload the parcels. For this task, the pupils will need to make sure that their forklift incorporates the Tilt Sensor.

**3. Create a map of a factory, and use the forklift to move parcels from one location to another.**

Ask the pupils to design the map of the factory or warehouse floor, and to define the best strategy for moving parcels one place to the other.

### ► Suggestion

An extra way of extending programming opportunities would be for the forklifts to move parcels from one factory to another. By doing this, the pupils would have to adapt their models and programs to new surroundings. This could even be used as an assessment opportunity.





## Create phase

### Design further solutions

Forklifts are not the only automated devices in a warehouse. Parcels are moved from one place to another using conveyor belts, robotic arms, and other sorting devices. To move boxes from one device to another, all of them have to work together and be coordinated.

#### 1. Create a sequence where a box is transported between at least three different devices.

Divide the pupils into teams and ask them to design different devices that could be used in a warehouse. Forklifts can still be used in this task, but some teams could design different devices.

#### ▶ Important

All of the teams should work with the same parcel design. Alternatively, the teams could work with a small ball.





## Share phase

### Complete the document

Make sure that your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

### Present results

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

### Suggestion

A good way to share this project is to organise an exhibition in your school to let pupils explain to visitors (other pupils, teachers, parents, general public) what they have achieved. Ask them to present their solutions as if they were a small company promoting their ideas.

## Project Extension 8

# Working Rover

This project is about collaborating with other groups to design and develop a multifunctional space rover for collecting soil samples.





## Curriculum links and Overview

### Main Learning outcomes

During this project, pupils will:

- Work with various forms of input
- Control or simulate physical systems
- Work with variables

### Supporting Learning Outcomes

During this project, pupils will:

- Design programs that accomplish specific goals
- Write programs that accomplish specific goals
- Use sequence in programs
- Work with various forms of output
- Debug programs that accomplish specific goals
- Use repetition in programs
- Use selection in programs
- Use logical reasoning to explain how some simple algorithms work
- Use logical reasoning to detect and correct errors in algorithms
- Create a range of programs, systems, and content that accomplish given goals, including collecting, analysing, evaluating, and presenting data and information
- Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content
- Use technology safely, respectfully, and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact

### Overview

In order for this project to be feasible, as it is an extension of an open project to which there are no specific models associated, it is suggested that pupils look at the Design Library to find inspiration for what they want to build.

### Adjustments required

Building inspiration for the model used in this project is provided in the Design Library of the software.

### Keywords

The programming focus is on input, simulating physical systems, and variables, but other concepts will be covered, particularly collaboration.



This project expands on the “Space Exploration” Open Project in the LEGO® Education WeDo 2.0 Curriculum Pack.



## Quick glance: Plan this WeDo 2.0 extension project

### ► Important

This extension project is planned to follow the Space science project. This is the easiest and most relevant way to implement this programming extension in a science context.

#### **Preparation: 30 min.**

- Read about this project extension so you have a good idea of what to do.
- Define how you want to introduce this project: Use the video provided in the space project in the WeDo 2.0 Software, or use material of your choice.
- Determine the parameters for presenting and producing the document.
- Make sure timing allows for expectations to be met.

#### **Explore phase: 10-15 min.**

- Connect pupils to the extension tasks using the Max and Mia story.
- Hold a group discussion.

#### **Create phase: 45-60 min.**

- Ask the pupils to adapt their model for the task if required.
- Let them program the model to solve the task.
- Allow them time to test the different combinations.

#### **Create more phase (optional): 45-60 min.**

- You can use this extra layer of the project for differentiation or for older students.

#### **Share phase: 45 min. or more**

- Make sure pupils document the different solutions they come up with.
- Ask the pupils to create their final presentations.
- Find different ways to let the students share their results.
- Ask the pupils to present their projects.

#### **Assessment opportunities**

If specific feedback can be provided for pupils to improve their computational thinking, focus could be around:

- How the design of their rover helped them to solve the problem.
- How the program they created helped them to solve the problem.
- What elements of their process were good and which ones they could improve.
- What elements of their document and presentation helped explain the process and the results.



## Explore phase

Max and Mia have some new ideas for a multifunctional space rover. They have made a list of different functions that could be useful, such as collecting extraterrestrial bugs or seeds from the soil.

They would like to have a longer list of functions that would be useful when exploring.

Engage in a discussion with the class and allow them to come up with some different and innovative rover functions. Ask each pupil to come up with one or two ideas and put them into categories such as funny, realistic, or creative.

### Suggestion

You could show the pupils some video footage of space rovers collecting samples. Alternatively, the pupils could use their devices to search for videos and information online. It's a good idea for you to research online images and videos first, so you can provide the pupils with a list of key search words. These could include:

- rock sample
- collecting
- robot
- space
- space rover
- curiosity
- Mars rover
- etc.



## Create phase

Let the pupils build solutions based on the inspirational models in the Design Library or from their own imagination.

Ask the pupils to develop solutions to this task:

### 1. Build a multifunctional space rover that can:

- Drive on the surface of a planet
- Collect rock samples
- Deliver rock samples to a place where they can be examined

The rover will be sent to the planet unmanned, so it will need to be operated by remote control. In order to solve this problem, you will have to collaborate with other groups to design, build, and program a rover that can solve the task.

Once the pupils have shared the task between teams (it is easier if one team is responsible for the driving base and one team for the robotic arm), let them build different prototypes. They will probably have to modify their builds so they can be joined together to create the final model.

There will be more than one solution to the task. The pupils should explore various options before deciding which one to build, program, and document. There are clear differentiation opportunities within this task.

There are no suggested programming solutions for this activity.

It's best to introduce this activity to the pupils when they have completed several other projects. By that time, they will be familiar with the models and will be able to make informed choices about which base models to choose, as well as having a good understanding of the programming software.

### ▶ Important

It's a good idea for the pupils who build, for example, the robotic arm, to have responsibility for programming that item. The groups will need to test and retest their programming to ensure that all of the programs work together.



## Create phase





## Share phase

### **Complete the document**

Make sure that your pupils document their work, including photographs, videos, and screenshots of their programs. When the extension project is completed, ask the pupils to create their final reports and to present their solutions.

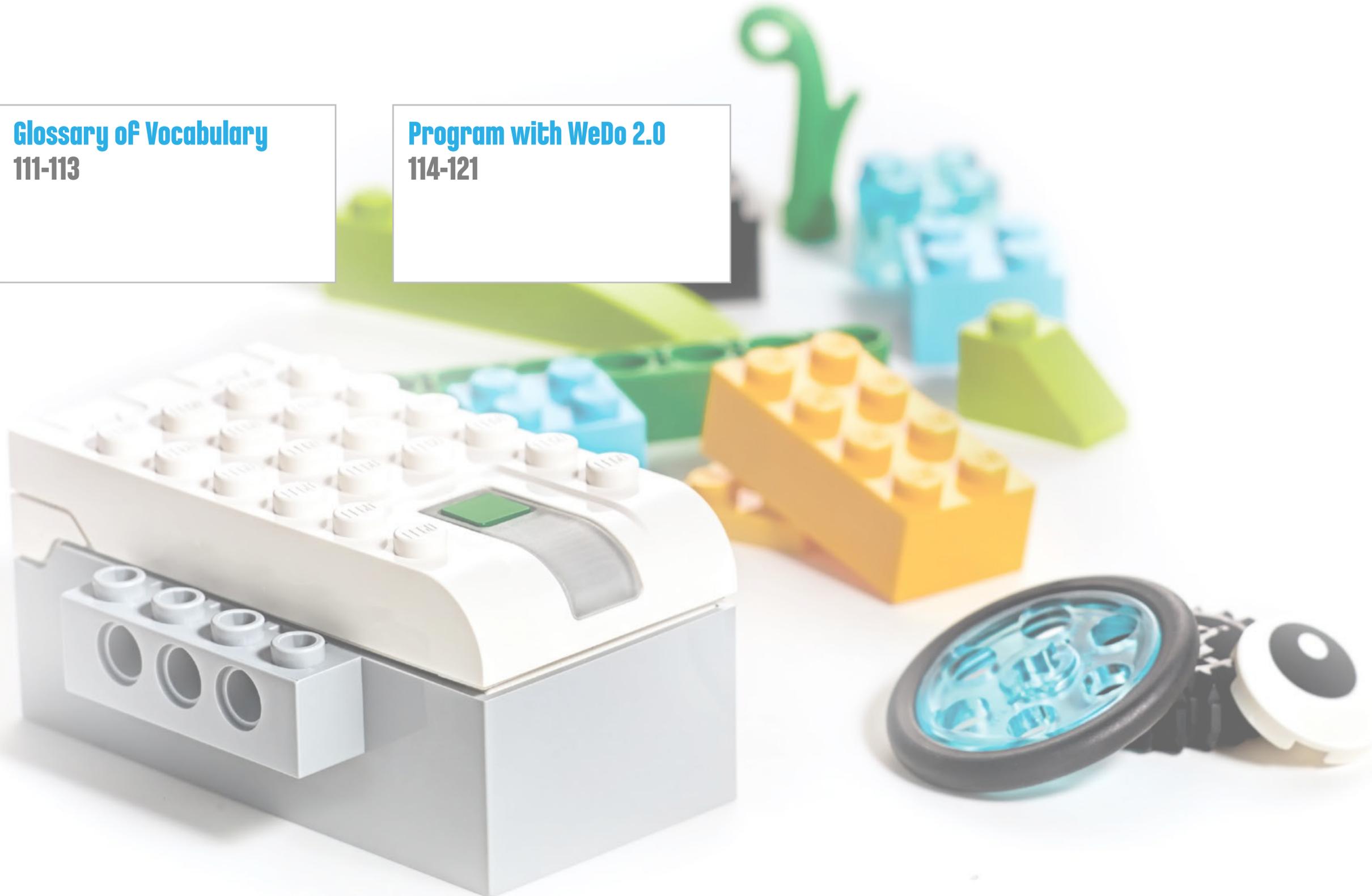
### **Present results**

Use different ways to let the pupils share their reflections and what they have learned from these experiences.

# LEGO® Education WeDo 2.0 Toolbox

**Glossary of Vocabulary**  
111-113

**Program with WeDo 2.0**  
114-121



# Glossary of Vocabulary

In this chapter, you will find a list of common programming terms and pupil-friendly definitions, which will help you explain concepts to the pupils as they occur in the projects.





# Using the correct word

## Algorithm

A set of instructions a computer uses to get something done. We use algorithms in our daily lives, such as when getting dressed or making a cup of tea, to ensure we do everything in the right order.

## Debug

The process a computer programmer goes through to find mistakes and correct them. We often debug our written work when we look for spelling mistakes and missing punctuation, so that it can be read correctly.

## Decomposing

When we are solving problems, or creating large algorithms, it is sometimes easier to break them down into smaller parts so that they are easier to contemplate.

## Execute

When a written program string is activated, it is often described as the execution of the program.

## Input

Input is data that a computer or device receives. It can be in the form of tapping a button, or through the use of sensors. A sensor is an example of an input device.

## Linear sequence

The actions you program will occur in the order they are in the program.

## Loop

A computer programmer can use specific program blocks or written code that tells an algorithm to keep repeating. This means it will keep doing the same thing until it is told to stop, rather than just doing it once.

## Output

Output is data that a computer or device sends. In WeDo 2.0, this could be: sound, lights, display, or turning motors on and off. A motor is an example of an output device.

## Parallel sequence

Sometimes, we may want two or more program strings to execute at the same time, controlled by one action. We can do this in WeDo 2.0 by using the Start on Key Press block.

## Program

A program tells the computer, step by step, exactly what you want it to do. You can choose how you want the computer to do something by using algorithms. A computer cannot work without a program. A program can be composed of more than one programming string.

## Program String

This is a sequence of programming blocks, joined together to make an algorithm.

## Repetition sequence

Another word for loop. A computer programmer can use specific program blocks or written code that tells an algorithm to keep repeating. This means it will keep doing the same thing until it is told to stop, rather than just doing it once.

## Selection

A computer programmer must select the best program blocks for a particular algorithm, depending on what they want it to do. There is often more than one way to create a program, so selecting the best block for the job is important to make the program efficient.



# Using the correct word

## **Sensor**

*A sensor detects or measures something (such as distance) and converts it into a digital signal. You will be using the LEGO® Education WeDo Motion and Tilt sensors. When a sensor is used in an algorithm it will tell the program when to execute a particular task, such as turning on a motor.*

## **Sequencing**

*It is important for a programmer to put program blocks or written code in the correct order. Sequencing is used to create a structure where one action in the algorithm leads to the next action.*

## **Variable**

*As in science, a variable is an element, feature, or factor that can be changed or varied in some way. When a variable is changed, the program will be affected, for example, making a motor run faster or playing a sound louder.*

# Program with WeDo 2.0

In this chapter, you will find help related to programming with LEGO® Education WeDo 2.0 Software, including

- Program strings description
- Program types
- Programming blocks description





## Introduction to a WeDo 2.0 program string

When pupils want to give life to their models, they will drag and drop blocks onto the Programming Canvas. Your pupils will be creating program strings. They can create multiple program strings on the canvas, but each has to start with a Start Block.

### Here are some important terms to use:

1. Start Block  
A Start Block is required to execute a program string.
2. Programming block  
Programming blocks are used in the WeDo 2.0 Software to build a program string. Blocks with symbols are used instead of text code.
3. Program string  
A program string is a sequence of programming blocks.





## Different types of programming strings

When pupils are exploring programming for the first time, they might line up as many blocks as possible on the programming canvas. They will want these blocks to execute an idea they have in mind. There are some general principles that can be explained and explored that will help them in bringing their ideas to life.

### 1. Linear sequence

A linear sequence is when blocks are placed one after the other in a linear fashion. The LEGO® Education WeDo 2.0 Software will then execute one action after the other in the order that the blocks have been placed.

### 2. Parallel sequence

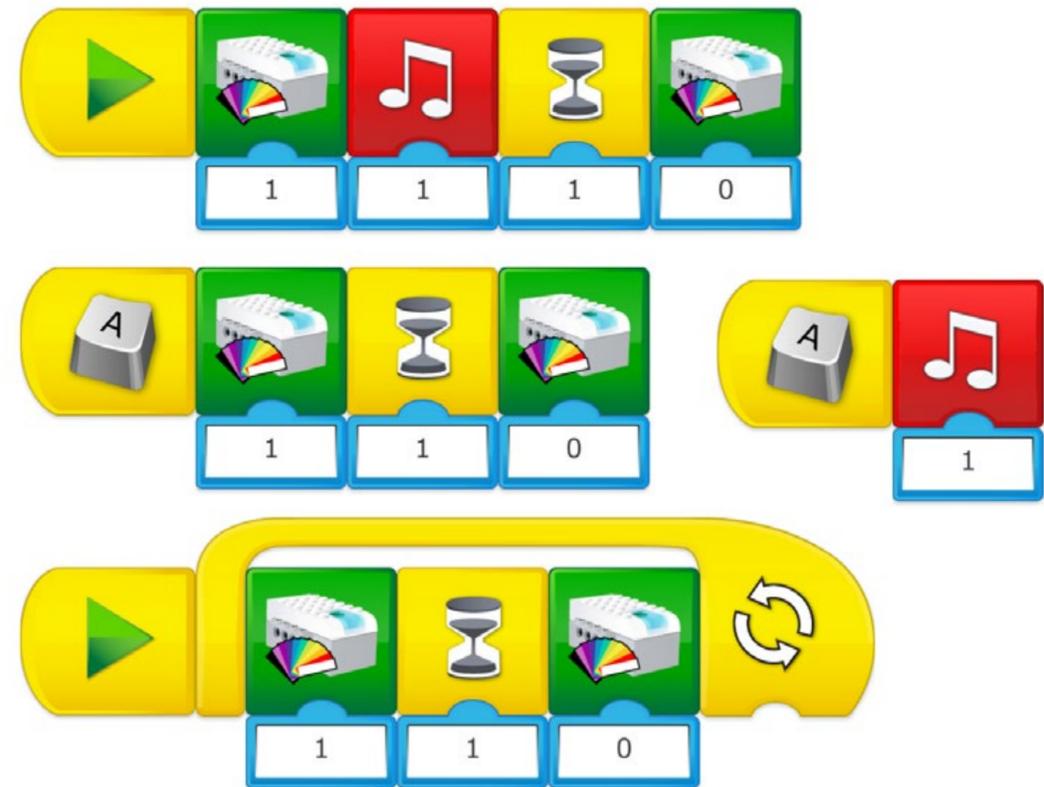
A parallel sequence should be used when pupils want to perform two or more actions simultaneously. Actions should then be placed on different program strings and be executed at the same time, using various techniques available in WeDo 2.0.

### 3. Repeat sequence

A repeat sequence can be used where an action needs to be repeated again and again. Using a Repeat Block will prevent the pupils from adding more blocks to a linear sequence. A repeat sequence cannot be used all the time, but when it is used, it is often considered as “more efficient” than a linear sequence, as pupils would use less programming blocks for the same action. A repeat sequence can allow pupils to repeat an action infinitely, which cannot be achieved using a linear sequence.

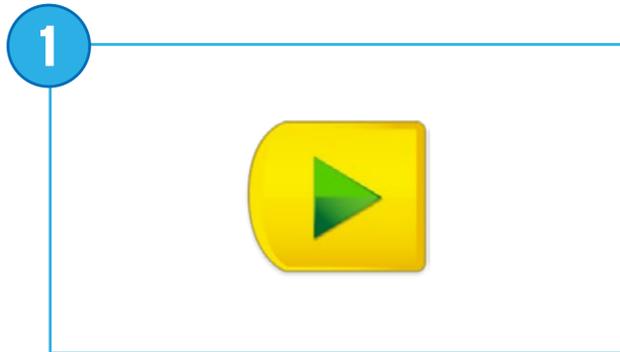
### ▶ Suggestion

Tell your pupils to plan their programs in advance. This will help them when deciding the order in which the program actions take place. They could do this using the Documentation tool, writing the actions they wish to program, step by step. They could also use a mind map of their own to decide if they should use a linear sequence, a parallel sequence, or a repeat sequence.

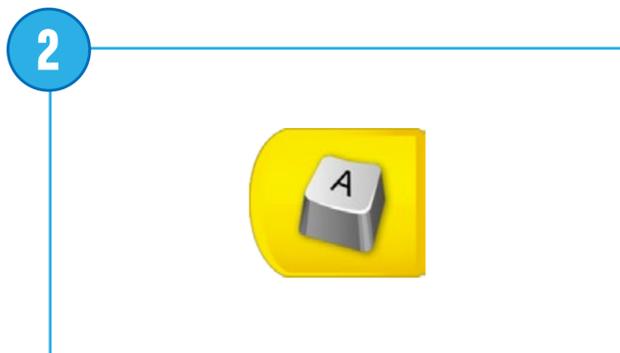




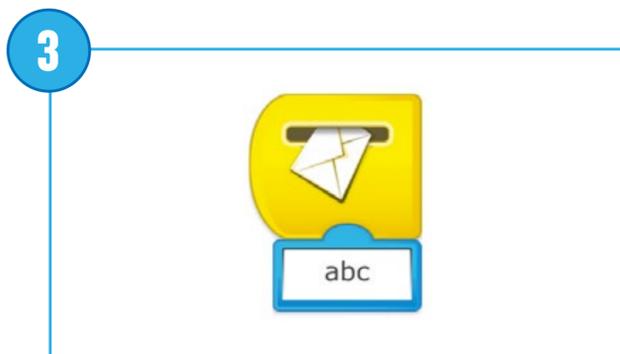
# Description of each programming block



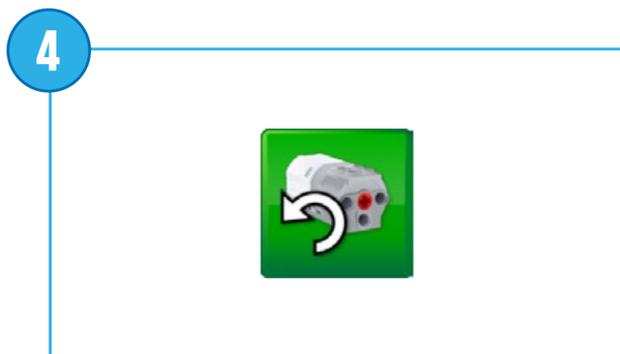
**Start Block**  
Always placed at the beginning of a programming string, press on it to start the program string you have written.



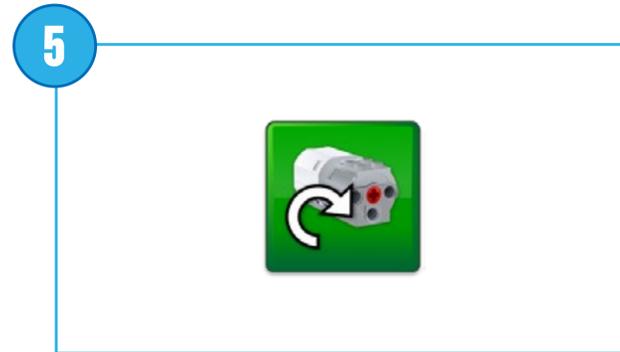
**Start On Key Press Block**  
Always placed at the beginning of a programming string, press on it or on the right letter on the keyboard to start the program string you have written. All the program strings with the same letter will start at the same time. To change the letter of activation, long press in the block to get access to the keyboard.



**Start On Message Block**  
Always placed at the beginning of a programming string, it will wait for the correct message to start the program string you have written.



**Motor This Way Block**  
Sets the motor to turn the axle in one direction.



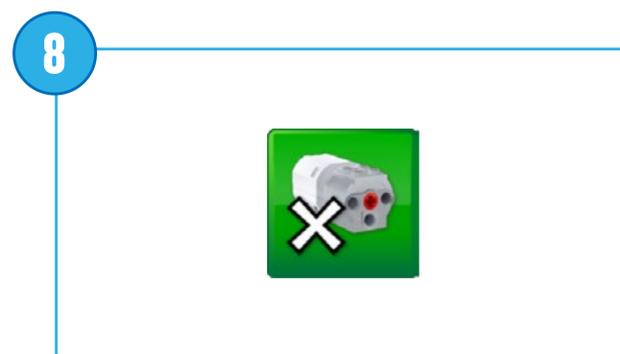
**Motor That Way Block**  
Sets the motor to turn the axle in another direction.



**Motor Power Block**  
Sets the motor power to the required level. The level can be set with a numeric input from 0 to 10.



**Motor On for Block**  
Activates the motor for a chosen amount of time. The amount of time can be set with a numeric input, using whole or decimal numbers.



**Motor Off Block**  
Deactivates the motor.



# Description of each programming block

**1**



### Light Block

Lights up the LED on the Smarthub in a specific colour. The colour can be changed with a numeric input between 0 and 10.

**2**



### Display Background

Use this block to display an image taken from a list available in the software. You can choose an image with a numeric block from 1 to 20.

**3**



### Display Block

Use this block to open the display area on the software screen. Entered numbers or text will appear in the display area.

**4**



### Add to Display

Use this block to mathematically add a quantity to the number currently shown in the display. Simply enter the number you wish to add.

**5**



### Subtract from Display

Use this block to mathematically subtract from the number shown in the display. Simply enter the number you wish to subtract.

**6**



### Multiply by Display

Use this block to mathematically multiply the number shown in the display. Simply enter the number you wish to multiply by.

**7**



### Divide by Display

Use this block to mathematically divide the number shown in the display. Simply enter the number you wish to divide by.

**8**

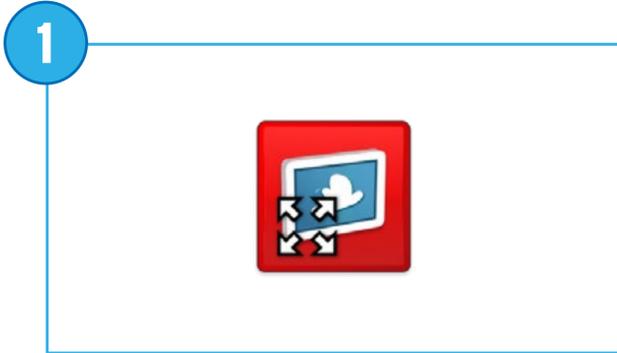


### Display Closed

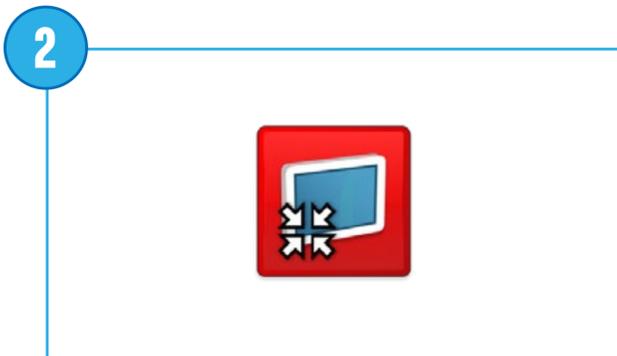
Use this block to close the display area on the software screen.



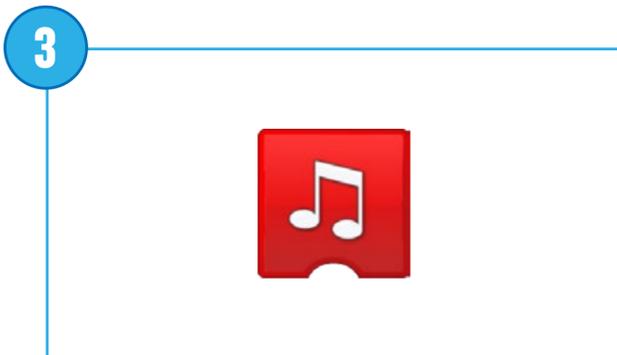
# Description of each programming block



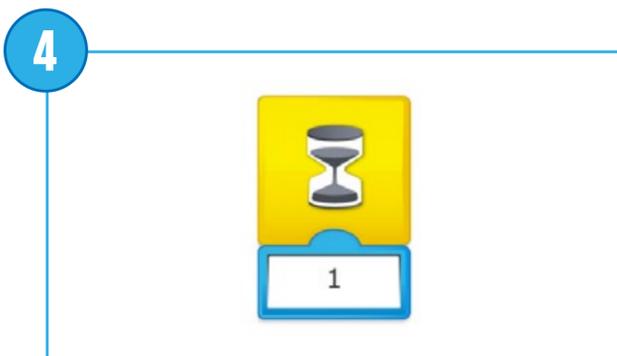
**Display Full Size**  
Use this block to set the display area to full size.



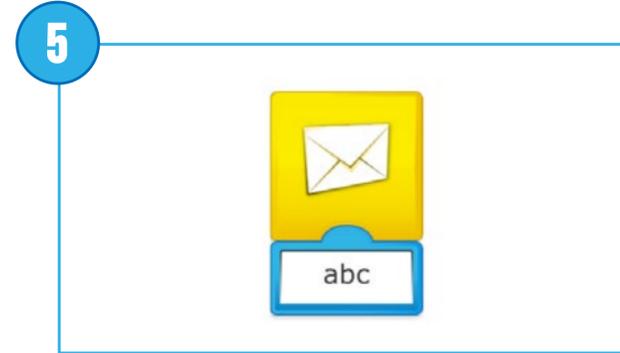
**Display Medium Size**  
Use this block to set the display area to medium size.



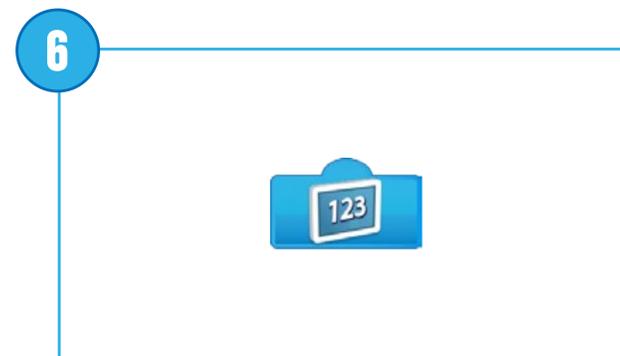
**Play Sound**  
Plays a sound. The sound is taken from a list available in the software. You can choose a sound with a numeric input from 1 to 20. Choose sound number 21 to play a sound that you have recorded yourself.



**Wait For**  
Use this block to tell the program to wait for something to happen: it can wait according to a set time or for an input from a sensor. This block always requires an input to work properly.



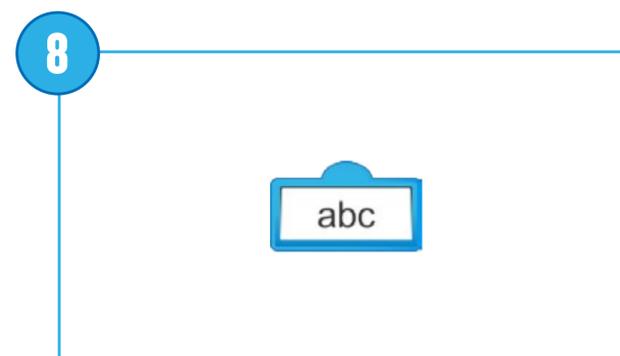
**Send Message**  
Sends a message to the programming canvas. Every Start On Message Block with the same message will be activated. The message can be in the form of text or numbers.



**Display Input**  
Inputs the numeric value shown on the display area to a block.



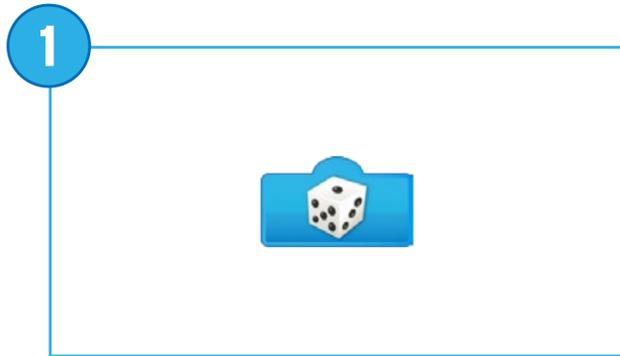
**Number Input**  
Inputs a numeric value to a block.



**Text Input**  
Inputs a text value to a block.

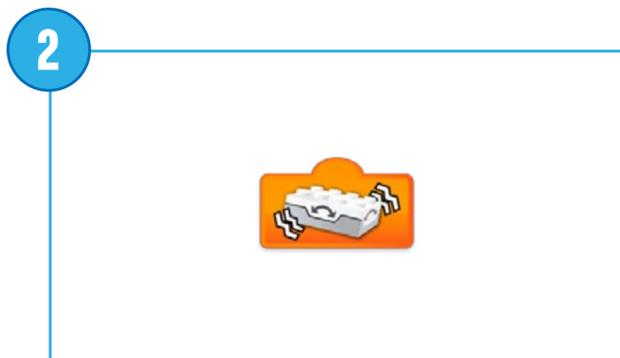


# Description of each programming block



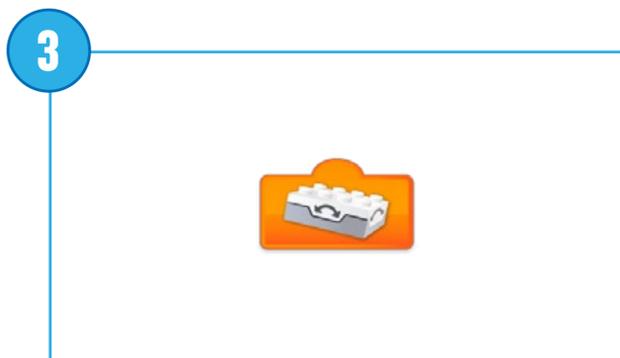
### Random Input

Inputs a random value between 1 and 9, to a block.



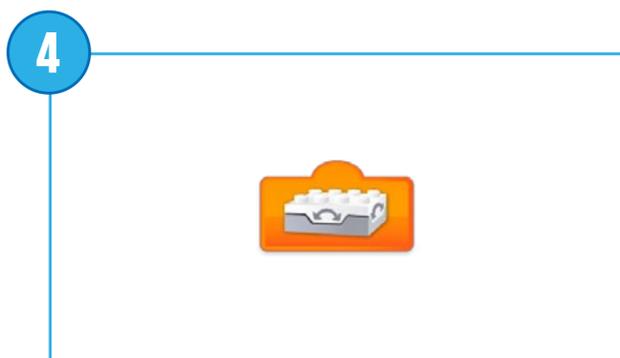
### Any Tilt

Inputs the Tilt Sensor state of Any Tilt to a block.



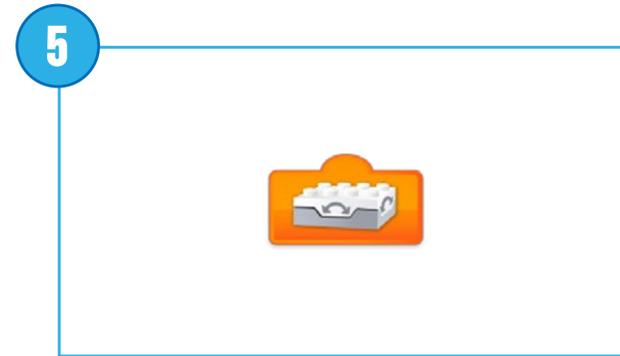
### Tilt Down

Inputs the Tilt Sensor state of Tilt Down to a block.



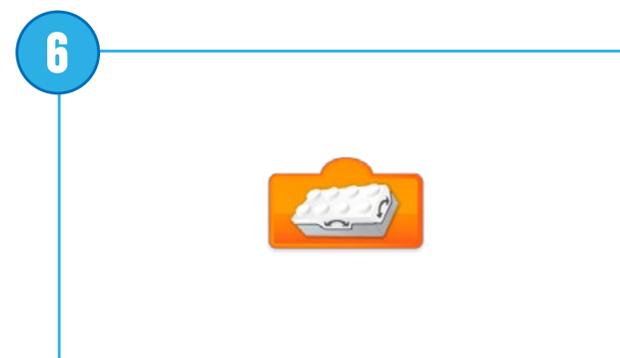
### Tilt Sensor Input/No Tilt

Inputs the numeric value generated by the Tilt Sensor to a block. Also inputs the Tilt Sensor state of No Tilt to a block.



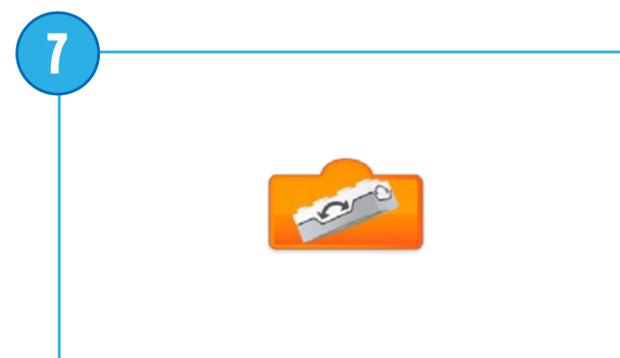
### Tilt That Way

Inputs the Tilt Sensor state of “tilt one way” to a block.



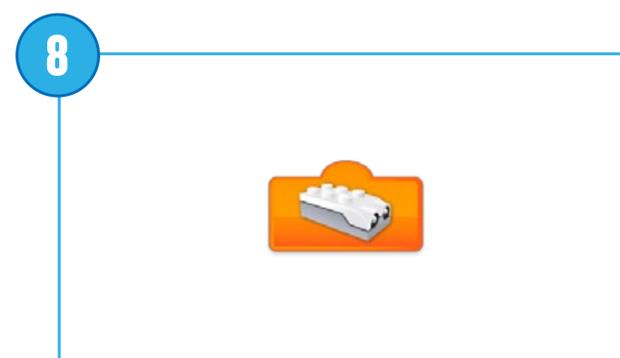
### Tilt This Way

Inputs the Tilt Sensor state of “tilt the other way” to a block.



### Tilt Up

Inputs the Tilt Sensor state of Tilt Up to a block.



### Distance Sensor Input

Inputs the value detected by the Motion Sensor to a block.



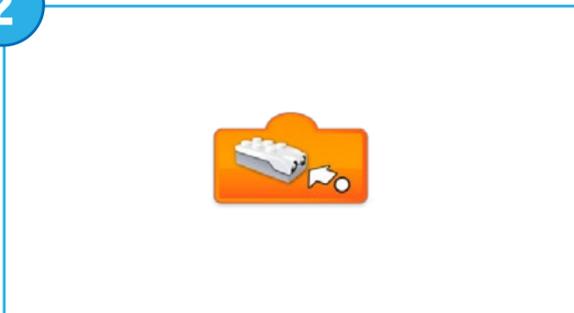
# Description of each programming block

1



**Any Distance Change**  
Inputs the Motion Sensor state of Any Distance Change to a block.

2



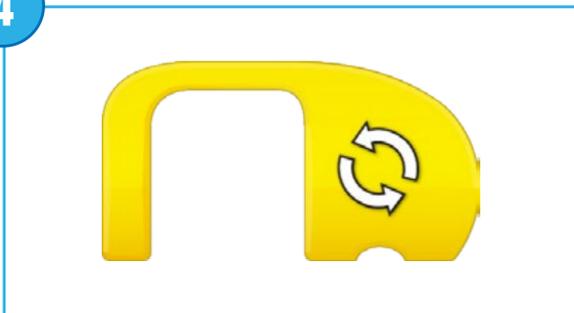
**Distance Change Closer**  
Inputs the Motion Sensor state of “decreasing distance between the sensor and an object” to a block.

3



**Distance Change Further**  
Inputs the Motion Sensor state of “increasing distance between the sensor and an object” to a block.

4



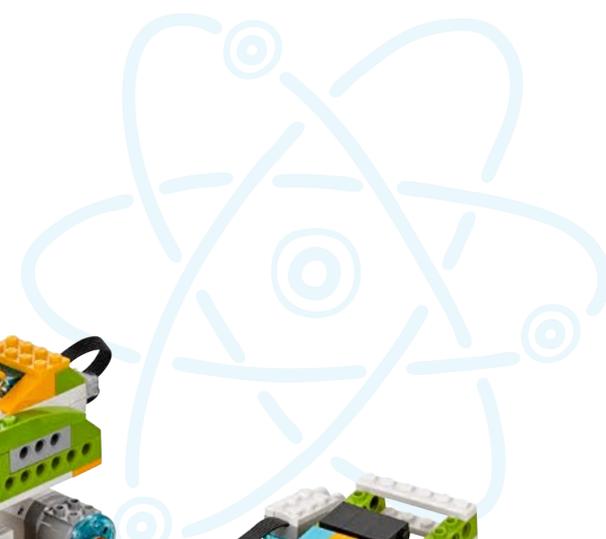
**Repeat Block**  
Use this loop to repeat actions. Blocks placed in the Repeat Block will be looped.

5



**Bubble**  
Use the bubble to insert comments in your program. This is not a programming block.

# LEGO® Education WeDo 2.0



[LEGOeducation.com](http://LEGOeducation.com)

LEGO and the LEGO logo are trademarks of the/sont des marques de commerce du/son marcas registradas de LEGO Group.  
©2016 The LEGO Group.

