robo™ wunderkind

Summer Camp

with Robo Wunderkind Robotics Kit.

Imagine, Build, Code, Play!

Introduction

Welcome to the Robo Wunderkind summer camp curriculum! We are happy to introduce you to the intensive robotics and coding program with Robo Wunderkind Robotics kit. This camp will bring your students from little or no technical experience to confident coders and inventors using the Robo Wunderkind visual programming language. They'll learn the basics as well as many more advanced coding tricks, building and programming many custom robots and learning to express themselves creatively. Our ready-to-use curriculum is made to fully support you during the preparation and entire course of the camp with Robo Wunderkind. You can join your students in playing with Robo: learning, coding, and imagining together!

Included in this ready-to-use curriculum:

- All the **key information** and **details** to organize the summer camp, as well as formulated **Learning Outcomes** and **Concepts Overview**;
- **10** scaffolded and easy to follow **lesson plans**;
- **6 additional activities** to increase the complexity of a lesson or extend the lesson's time;
- **Supporting materials** to make your teaching more comfortable.

Table of Contents:

robo™ wunderkind

Key Information

robo™ wunderkind

Topic: STEAM subjects | **Student's age:** 6-8, 9-12 | **Group Size:** 6 – 12 students

Duration:

- **Day 1-4:** Two projects per day; each project is 1.5 hours; 12 hours of programming in total;
- **Day 5:** Preparation and presentation of the final project for the parents; 3 hours in total;

Students' Age:

We recommend forming two groups of students, by age: age 6-8 or age 9-12.

Recommended Prior Knowledge: Students do not need any prior knowledge. They will learn the basic terminology and the principles of robotics and programming using Robo Wunderkind robotics kit.

Complexity: Each lesson includes the **basic level** as well as a possible modification for more advanced students.

Additional activity: This makes it possible to adapt the complexity of the lessons to the personal needs of your students; from very simple coding to more complex projects and challenges.

Materials Required:

- Robo Wunderkind robotics kit(s);
- Tablet(s);
- Some materials to customize robots and create an environment: Lego™ bricks, colored paper, cardboard etc.;
- Supporting materials: printable coding icons and Robo Wunderkind Modules, Key Vocabulary and Engineering Design Process cards.

Short Program Overview

robo™ wunderkind

Days		Complexity	Concepts
Day 1	I. Meet Robo!	☆	Robotics, Engineering, Controls and Mechanical design
	II. Robo Explores its Surroundings	☆	Programming, code; Sequential Logic; RW Visual Based Programming: Action, Connection, Loop
Day 2	I. Robo Transforms into Devices	☆☆	Electronic device, Mechanical design, Problem-solving; RW Visual Based Programming: State, Parallel Execution
	II. Robo is a Flashlight	☆☆	Algorithm, Button, Mechanical, and Code Design, RW Visual Based Programming: Transition, Condition
Day 3	I. Robo is a Smart Alarm Clock	☆☆☆	Smart devices, Mechanical, and Code Design, RW Visual Based Programming: Transition, Condition
	II. Robo is a Smart Pet	☆☆☆	Smart devices, Sensor, Mechanical and Code Design, RW Visual Based Programming: Transition, Condition
Day 4	I. & II. Project with Robo for the Final Presentation	☆☆☆	Engineering Design Process, Mechanical and Code design
Day 5	I. & II. Final Presentation	☆☆☆	Engineering Design Process, Mechanical and Code design

Robotics and Computer Science Concepts Covered in Robo Wunderkind Camp Program

robo™ wunderkind

Concepts	Day 1		Day 2		Day 3		Day 4		Day 5	
	Project 1	Project 2	Project 3	Project 4	Project 5	Project 6	Project 7	Project 7	Final presentation	
Robotics										
1. Robotics, Engineering	+	+	+	+	+	+	+	+	+	+
2. Electricity:										
• Electrical Power	+	+	+	+	+	+	+	+	+	+
• Local Communication	+	+	+	+	+	+	+	+	+	+
3. Wireless Communication	+	+	+	+	+	+	+	+	+	+
4. Remote Control	+						+	+	+	+
5. Design Thinking Process:										
• Controls Design	+								+	+
• (Mechanical) Design	+	+	+	+	+	+	+	+	+	+
• Code Design		+	+	+	+	+	+	+	+	+
6. User Input: Using the Controls (Sounds, Light, Motors, Servo Controls)	+						+	+	+	+
7. Outputs: Functions of Modules										
• Outputs: Sounds	+	+			+	+	+	+	+	+
• Outputs: (RGB) Light	+	+	+	+	+	+	+	+	+	+
• Outputs: DC Motors	+	+	+		+	+	+	+	+	+
• Outputs: Servo Motor	+		+			+	+	+	+	+

Robotics and Computer Science Concepts Covered in Robo Wunderkind Camp Program

robo™ wunderkind

Computer Science										
1. Programming, Code		+	+	+	+	+	+	+	+	+
2. State-Machine Based Programming:										
• Action		+	+	+	+	+	+	+	+	+
• Connection		+	+	+	+	+	+	+	+	+
• State			+	+	+	+	+	+	+	+
3. Sequential Logic		+	+	+	+	+	+	+	+	+
4. Loop			+	+	+	+	+	+	+	+
5. Parallel Execution			+	+	+	+	+	+	+	+
6. User Input										
• Software Input: parameters of Actions	+	+	+	+	+	+	+	+	+	+
• Sensor Input: Button				+	+		+	+	+	+
• Sensor Input: Distance Sensor						+	+	+	+	+
• Sensor Input: Sound Sensor						+	+	+	+	+
7. Digital Literacy	+	+	+	+	+	+	+	+	+	+
8. Problem Solving			+	+	+	+				
9. Engineering Design Process							+	+	+	+
10. Presentation								+	+	+

Learning Outcomes related to Robotics and Computer science with RW Robotics Kit:

robo™ wunderkind

I. Robotics:

- Understand and follow the classroom rules of using technology – **Digital literacy**;
- Understand what **robots** are and their functions in everyday life; what **robotics** is;
- Understand what **electrical power** and **wireless communication** (Bluetooth) is and why robots need it;
- Understand how people **control robots**: the difference between **remote control and programming**;
- Understand what **Mechanical Design** and **Code Design** is and can use it to create a Robo-project.

Robo Wunderkind Robotics Kit:

- Know some of **Robo's Modules** and **Connectors**, and understand their functions – **Outputs**;
- Know what the **Main Block** is, understand its functions; can explain why it is necessary for every project;
- Understand and take into account the general **logic of building robots** with Robo Wunderkind robotics kit;
- Can combine modules to consider the functions required in order to **build a robot for concrete purposes** – use Mechanical Design;

Robo Live App:

- Know and can use the **Robo Live App:** can connect the Main Block, create a new project, add controllers and manage them;

II. Computer Science:

- Understand what **programming and program** is;
- Understand the terminology of **State-Machine Based Programming** such as **Action, Connection, State, Loop** and can use **Visual Based Programming language** of Robo Code App to create a simple program;
- Understand the difference between a **sequential logic** program and a program with **parallel execution**; can create both;
- Understand what parameters of Actions – **User Input** are and their function for the robot's performance;
- Understand and can follow the **Engineering Design Process** in order to create a Robo-project;

Robo Code App:

- Know and can use the **Robo Code App:** can connect the Main Block, create a new project; use the Menu;
- Recognize and can use the **Visual Based Programming language** in Robo Code App to create a simple program in order to **solve the set challenge(s)**.

Cognitive and Behavioral:

- Can pay **attention** to the information needed to complete certain tasks (Sustained, Selective, Alternating, and Divided attention);
- Can maintain **concentration** during the time allotted for completing tasks / projects;
- Can **memorize** information for short-term tasks, as well as for long-term periods;
- Can use **spatial thinking** in order to assemble robots which work in each certain way;
- Can use the **logical and algorithmic thinking** in order to control and code the robot in a specific way;
- Can **solve the problem** using critical thinking skills: set the goal, plan, action, reflect, recreate, evaluate, accept criticism;
- Can use **imagination and creativity** to create own project;
- Can **work in pairs / small groups** and use **social skills** to create a common project;
- Can **present** the created project to the class; can receive and give **constructive feedback** on the project.

Suggested Structure for a Project with Robo Wunderkind Robotics Kit

robo™ wunderkind

To Plan:

- Focus on the particular concepts in relation to RW Modules, Robo Live or Robo Code App;
- Objectives and Learning Outcomes; Key vocabulary;
- Printable supporting materials.

Activity Stages: 8 Steps

Lead-in *7 – 10 min*

1. **Activate students:** Draw on previous knowledge and personal experiences.
2. **Analyze:** Tell Robo's Story to connect with the students on an emotional level, **identify** the problem situation, discuss, and set the **project goal.**

Guided Activity *15 – 25 min*

3. **Get ready: Recall** some of the previous knowledge about the RW robotics kit and the Apps; **Connect** the Main Block to the tablet and create a new project.
4. **Learn by doing – build and program together:** Let students solve different challenges and gather knowledge through cooperative play and discussions.
5. **Sum up** new information before the independent activity.

Independent Activity *30 – 45 min*

6. **Make your own project:** Students work individually / in pairs / in small groups to create their own project.
   The teacher provides struggling students with one-on-one assistance. They can use different materials in order to create an environment and customize robots. Students can also cooperate to make a shared project with two or more robots.
   Presentation (optional): Students present their projects to the class and give constructive feedback to each other.

(*) ***Additional activity:*** *Additional task to* ***increase the complexity*** *of the lesson for more advanced students.*

Reflection & Feedback *8 – 10 min*

7. **Sum up** the learned information. **Receive feedback** about the complexity of tasks and activities.
8. **Clean up:** Teach students to take care of devices they use; turn the power off on the orange Main Block, take it apart, and put all modules of Robo back into the boxes; lock and carefully collect all the tablets.

Expected time of each lesson: 60 – 90 min

Day 1. Project 1: Meet Robo!

robo™ wunderkind

Concepts:
Robotics, Engineering, Controls and Mechanical Design

Complexity: ★☆☆

Robo's Story:

Today we have a special guest in our class! This is Robo, a smart robot that has come to summer camp to be our friend.

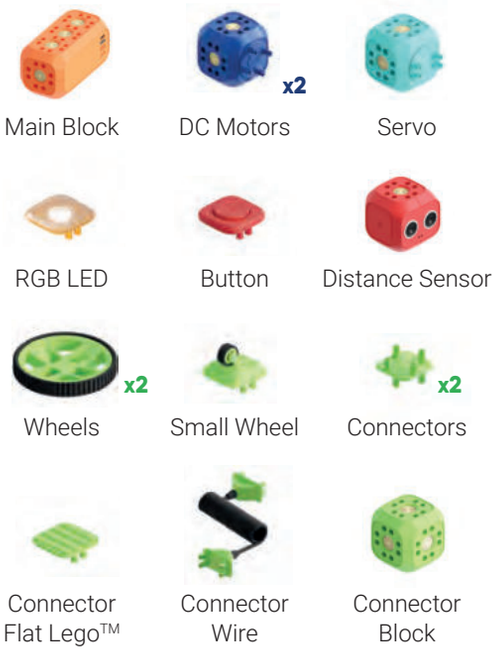
Project Goal:

We will become engineers in order to assemble and control our first robot!

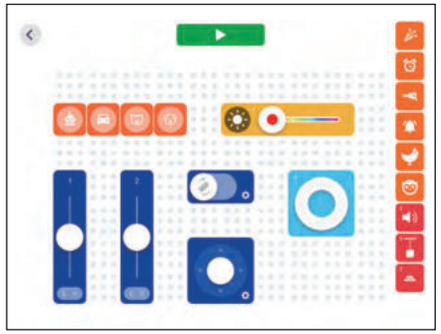
Day 1. Project 1. Meet Robo!

robo™ wunderkind

Modules:



Control:



Focus:

- **Robotics:** Robots and their functions in everyday life, remote control as a way of controlling robots.
- **Digital Literacy:** Rules for tablets / devices use.
- **Robo Live App:** Interface and logic.
- **Robo Wunderkind Robotics Kit:** Some of the Modules, Connectors, and their functions.

Objectives:

To assemble the robot using different Modules and Connectors, and to consider their functions.

Learning Outcomes:

- I know some of Robo's modules and their functions;
- I can explain why every robot needs the Main Block;
- I can comprehend how to combine modules to build a robot;
- I can use Robo Live App to control different modules.

Key Vocabulary:

- Robotics, Engineering, Robot;
- Modules: Main Block, Connectors, Disconnecting Tool.

Day 1. Project 1. Meet Robo!

robo™ wunderkind

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** Do you know what a robot is? Why do people create robots? Who has experience with using robots? How do people control robots? Who studies robots? **Discuss** the terminology: **Engineering, robotics, robot.**
2. **Analyze:** Tell **Robo's Story**, discuss, and set the **project goal** together with your students.

Guided Activity *15 – 25 min*

3. **Open** and **explore the boxes with Robo:** Briefly examine the modules.
4. **Discover the Robo Live App:**
   - **Discuss** the need of tablet to control Robo; hand out the tablets and discuss the **rules for tablets use;**
   - Switch on the **Main Block** and connect it to the **Robo Live App;** explore – My Robo screen, Coding lab, and New Project button;
   - Create a **new project** and explore Controlling screen, Controls, and Controls Menu.
   - Ask students to **connect different Modules** to the Main Block and see how they appear in the App; **disconnect** them; discuss the Connectors and Disconnecting tool. Use appropriate Controls to control Modules and discuss how they work.
   - Introduce the term: **Remote control.**
5. **Sum up** new information before the independent activity.

Note: *Students do not need to know all the modules by the end of the lesson. Rather, allow them to explore the Modules by touching, connecting, and controlling them to see how they work****. Only pay attention to the Main Block and its function.***

Independent Activity *30 – 45 min*

6. **Make your own project:** Build any variation of Robo and control it using the Robo Live App. Share the story of why you created this particular robot and how it will help you in your life. Use some materials to customize your Robo.

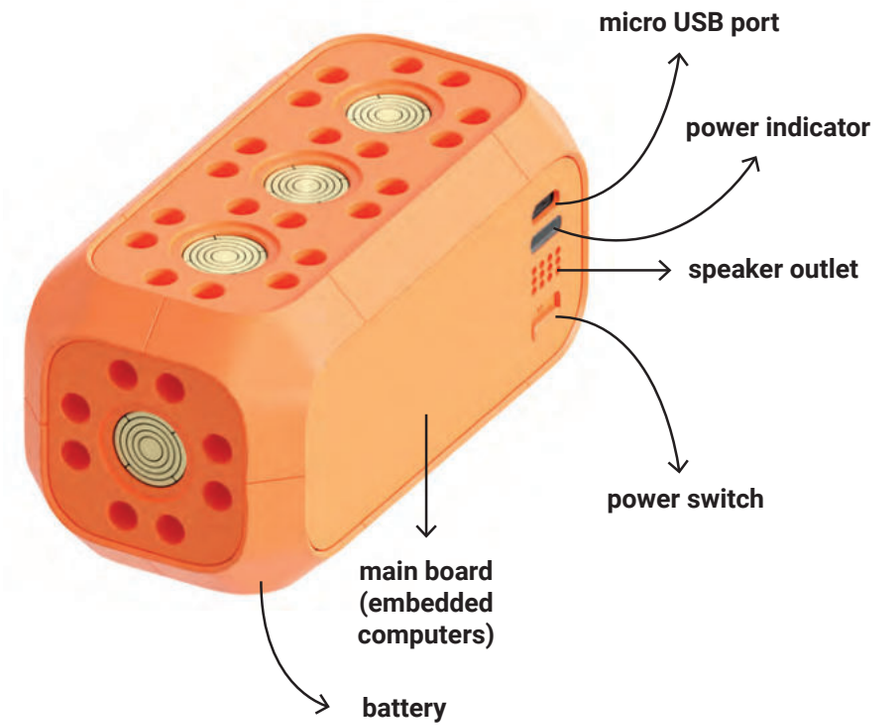
Reflexion & Feedback *8 – 10 min*

7. **Sum up:** Some of the Robo's Modules and their functions, the projects students have created using them. **Receive feedback:** Were the tasks easy or complex? Interesting or boring? Which part of the lesson was the most interesting? Why?
8. **Clean up:** Teach students to take care of the devices they use – RW Modules and the tablets.

Day 1. Project 1. ✱ Additional Activity

Goal: Explore **technical details** and **design** of the **Main Block.**

1. **Ask:** Which Module is the most important one? Why do you think so?
2. **Examine** the Main Block and **ask:** What do you see on the surface of the Main Block? What could be inside it?
3. **Conclude:**
   - The most important Module is the Main Block because it's the Robo's brain.
   - We can see the power switch, the small light — power indicator, speaker, USB port on the Main Block.
   - The battery and a small (embedded) computer are inside the Main Block.
4. **Show** the image of the **Main Block** and its technical details. **Discuss the function** of the power switch / power indicator / speaker / USB port / battery / small (embedded) computer.
5. **Conclude:** The Main Block powers the Modules; it can be connected to the tablet and receives the commands from it. Hence, the Main Block has to be in each project in order to make all other Modules work.



Day 1. Project 2: Robo Explores its Surroundings

robo™ wunderkind

Concepts:
Programming, code; Sequential Logic; RW Visual Based Programming: Action, Connection, Loop

Complexity: ★☆☆

Robo's Story:

Robo is a very curious creature who likes to explore the world! Robo wants to travel, but first Robo needs to learn how to interact with the objects — and also needs to learn how to drive!

Robots can do many things if you control them, but you can also program Robo to do things! There is a special world – the Robo Code App — where we can create code for Robo to perform.

Project Goal:

To help Robo go on its first journey, we will become programmers and create our first program code in the Robo Code App.

Day 1. Project 2. Robo Explores its Surroundings

robo™ wunderkind

Modules:


Main Block

DC Motors


RGB LED

Wheels


Small Wheel

Connectors

Program:


Sounds


Visuals


Constant Light


Blink


Movement


Drive


Turn

Focus:

- **Robotics:** Outputs – Sounds, RGB Light, DC Motors.
- **Computer Science:** the importance of coding in everyday life; User Input – Parameters of Actions.
- **Robo Code App:** Interface, Actions – Sounds, Visuals, Movement, Connection, Loop.

Objectives:

To create a simple sequential logic program that includes Sounds, Visuals, and some of the Movement: Drive and Turn actions and Connections between them, then modify it to form a loop.

Learning Outcomes:

- I understand what a program is;
- I recognize and can use the coding buttons in the Robo Code App to create a code;
- I recognize and can use Sound, Constant Light, Blink, Drive and Turn actions;
- I can connect two or more Actions together to make a simple sequential logic program.

Key Vocabulary:

- Sequential code;
- **Modules:** Main Block, (DC) Motors, (RGB) Light;
- **Robo Code App:** Programming screen, Action, Start Point, Connection; **Actions:** Sounds, Visuals – Constant Light and Blink actions; Movement – Drive and Turn actions.

Day 1. Project 2. Robo Explores its Surroundings

robo™ wunderkind

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** Who is our new friend? What did we create? How did we control our robots? How else can people control robots? What is the **difference between remote control and programming**? Have you tried to program before? What is programming? What is a program? **Discuss** the terminology: **Programming, code.**
2. **Analyze:** Tell **Robo's Story**, discuss, and set the **project goal** together with your students.

Guided Activity *15 – 25 min*

3. **Open** and **explore the boxes** and **recall** some of the Modules, Connectors, focus on the Main Block and its function.
4. Explore **the Robo Code App:**
   - **Connect** the Main Block to the Robo Code App and **discuss**: My Robo menu, Coding Lab screen, and New Project button. Ask students to share some information about their robots from the My Robo menu.
   - **Create a new project** and **discuss**: Programming screen, Actions Dock
   - **Program Sounds:** Teach your Robo to make sounds!
   - Learn about the **Start Point, Play button, Connection button** and Connection mode, **Trash Bin button** and Delete mode.
   - **Add (RGB) Light** to the build, then program **Visuals – Constant Light and Blink actions.** Teach your Robo to make different visual signals to light up the room or maybe interact with other robots!
   - **Add (DC) Motors** to the build, then program **Movement: Drive and Turn actions.** Teach your Robo to drive around.
   - **Play around:** connect all Sounds, Visuals, and Movement and **introduce a term – sequential code.** Modify code to form a **Loop**.
5. **Sum up:** What is our task to solve? Which skills and Modules will Robo need for its first journey? Which Actions did we program? How will they help Robo in its journey? Are we ready to travel with Robo?

Independent Activity *30 – 45 min*

6. **Make your own project:**
   - Decide where your Robo-traveler will go; create an environment and program Robo-traveler for this particular situation using all known Actions and Connections between them to create a sequential code. Use some materials to customize your Robo.
   - ***Presentation*** *(optional): Students present their projects to the class and give constructive feedback to each other.*

Reflexion & Feedback *8 – 10 min*

7. **Sum up:** What is programming, code; the Robo Code App interface and coding buttons; Sound, Constant Light, Blink, Drive and Turn actions and Connections, sequential code. **Receive feedback:** about the complexity of tasks and activities.
8. **Clean up:** Teach students to take care of the devices they use – RW Modules and the tablets.

Day 1. Project 2. ✱ Additional Activity

Goal: Discuss programming languages in general and specific **programming language for Robo Code App.**

1. **Ask:** Do all programs look the same? Why? What is the programming language? Do you know any of the programming languages?

2. **Discuss:** How does the programming language of Robo look like?
   - What is Action?
   - Where can we find these Actions?
   - Are all Actions the same color? Why do you think they are different colors?
   - Connect different Modules to the Main Block to find the logic.
   - How do we connect Actions into a sequential code?
   - What is important while drawing a Connection? How does the direction of the Connection affect a code?

3. **Show:** the screenshot of the Programming screen with the different Actions and Connections between them.

4. **Conclude:**
   - There are colorful Actions— coding icons which look like bubbles; these are commands for Robo to perform.
   - We can find all the Actions in the Actions Dock.
   - Actions are different color because they are connected to the functions of different Modules. There are 4 types of Actions in the Actions Dock: Sounds are orange as the Main Block, Visuals are yellow as the (RGB) Light, Movement is blue as Motors, and Specials are purple. We can program a certain Action only if the module which corresponds to this Action is assembled to your Robo's Main Block.
   - To create a sequential code we need to draw the Connection(s).
   - We have to pay attention to the direction of Connection(s) we draw; they show the order in which the Actions will be performed by Robo. If the order is wrong, there will be a mistake in the code.

Day 2: Project 3. Robo Transforms into Devices: Robo-fan, Robo-lamp

robo™ wunderkind

Concepts:
Electronic Device, Mechanical design, Problem solving; RW Visual Based Programming: State, Parallel Execution

Complexity: ★★☆

Robo's Story:

Robots are created by people to help them with any task which is too complex, too dangerous, or simply too boring to do ourselves. Can we also transform our Robo into a device to help us with a task?

Project Goal:

Today we will transform Robo into different devices to help us in our everyday life.

Situation #1: Robo-fan

Imagine a summer day when the classroom is getting too hot. How can we cool down the air? Which device would we use?

Situation #2: Robo-lamp

Imagine that you are sitting at home on a rainy day and want to make your room cozy and bright with colorful lights. Which device could you use?

Day 2: Project 3. Robo Transforms into Devices: Robo-fan, Robo-lamp

robo™ wunderkind

Modules:


Main Block


DC Motor


Servo


RGB LED


Wheel


Connector Block


Connector

Program:


Sounds


Visuals


Constant Light


Blink


Movement


Motor


Servo

Focus:

- **Robotics:** Outputs – Sounds, RGB Light, DC, and Servo Motors.
- **Computer Science:** User Input – Parameters of Actions; Parallel Execution.
- **Robo Code App:** Sounds, Visuals, Movement.

Objectives:

To create a program with a sequential code or a parallel execution that includes Sounds, Visuals, Movement.

Learning Outcomes:

- I know what an electronic device is;
- I can identify a problem and come up with a solution for it;
- I can combine modules to build a Robo which can perform the set task;
- I can use the Robo Code App to create a new project;
- I can create a simple sequential logic program or a program with parallel execution that includes Sounds, Visuals, and Movement.

Key Vocabulary:

- Electronic Device, Problem-solving process;
- **Modules:** DC Motor, Servo Motor;
- **Robo Code App:** Coding buttons such as Start, Stop, Trash Bin, and Connection button, Start Point, Action, Connection, Loop, State.

Day 2: Project 3. Robo Transforms into Devices: Robo-fan, Robo-lamp

robo™ wunderkind

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** What did we create last time? What did we program last time and why? Which modules did we use and why?
2. **Analyze:** Tell **Robo's Story**. **Ask:** What is an electronic device? Which devices do we use in everyday life? How do they help us?

Guided Activity *15 – 25 min*

3. **Solve the task #1** – Build and Program a **Robo-fan:** Guide students through the problem-solving process.
   - **Identify:** Read aloud **Situation #1**, **identify** the problem situation, and **come up** with a theoretical **solution**.
   - **Plan:** Decide which Modules and Coding Actions you will need for the project and why. Make a step-by-step plan: **Algorithm**.
   - **Build a Robo-fan:** Try different builds. **Mechanical design**, compare the DC Motor to the Servo Motor. Use the Robo Live App to see how Modules work and how different designs influence Robo's performance. Choose the best solution.
   - **Program a Robo-fan:** Explore the **Motor 1 action** and **Servo action** and their settings; compare them. How they are similar or different?
   - **Reflect:** Discuss the process and the final solution; pay attention to the DC and Servo Motors and Actions.
4. **Solve the task #2** – Build and Program a **Robo-lamp:** Let students practice the problem-solving process.
   - **Identify:** Read aloud **situation #2**, **identify** the problem situation and come up with a theoretical **solution**.
   - **Plan:** Decide which Modules and Coding Actions you will need for the project and why. Make a step-by-step plan: **Algorithm**.
   - **Build a Robo-lamp:** Try different builds. **Mechanical design**, use the Robo Live App to see how Modules work and how different designs influence Robo's performance. Choose the best solution.
   - **Program a Robo-lamp:** Learn about the **Parallel Execution**. Program Visuals, Sounds, and Movement as sequential code; then, modify them into one **State**. Discuss the difference in Robo's performance. Introduce the terms: State, Parallel Execution.
   - **Reflect:** Discuss the process and the final solution; pay attention to the **Parallel Execution** and **State**.
5. **Sum up:** What is the Mechanical design and why it is important in the projects? What is the difference between the sequential code and parallel execution?

Independent Activity *30 – 45 min*

6. **Individual or group activity:**
   - Build and program a **Robo-device to help you in everyday life**. Use the Main Block, RGB Light, DC and Servo Motors. Include both sequential logic and parallel execution in a code. Plan, build, and program to carry out the project. Use some materials to customize your Robo.
   - ***Presentation*** *(optional): Students present their projects to the class and give constructive feedback to each other.*

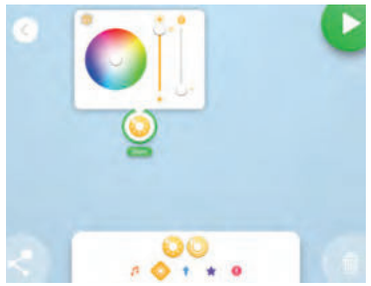
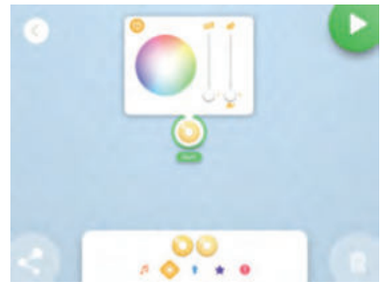
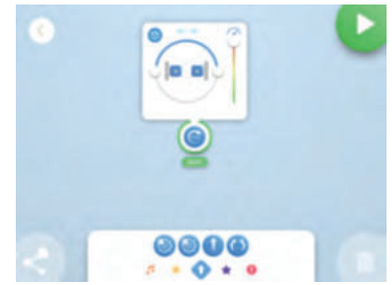
Reflexion & Feedback *8 – 10 min*

7. **Sum up:** Motor 1 and Servo actions and their settings, Parallel Execution, State, which Action can form a State. **Receive feedback:** about the complexity of tasks and activities.
8. **Clean up:** Teach students to take care of the devices they use – RW Modules and the tablets.

Day 1. Project 3. ✱ Additional Activity

Goal: Learn about the **Random setting.**

1. **Find the Random setting button** in the Constant Light action settings.
   - **Try out** the Random setting for Constant Light action and discuss how it influences the Robo performance.
   - **Ask:** What does "random" mean? Who generates the setting if it is random?
   - **Find** the other Actions which have the Random setting option and try them out: **Constant Light, Blink, Turn and Servo actions.**


2. **Program the Random setting** for different Actions:
   - **Program the Random setting for Visuals:** Constant Light and one Blink actions in different variations;
   - Compare them to the Actions with set color and discuss;
   - **Program the Random setting for Movement:** Turn and Servo actions in different variations;
   - Compare them to the Actions with a set angle;
   - Compare Visuals and Movement with the Random settings – which parameter is controlled by it?

3. **Create a small project:** Come up with an idea of why you need to program the Random setting.

Examples of the projects:

- **Visuals:**
  Robo's Story: Robo uses different colors to express its mood.
  Game: Robo makes encrypted light signals for other robots; you need to remember them and try to repeat.
- **Movement:**
  Robo's Story: Robo shakes its head while dancing.
  Game: Robo looks to different angles; you have to guess which way it will turn its head next time.

Day 2: Project 4. Robo is a Flashlight

robo™ wunderkind

Concepts:
Algorithm, Button, Mechanical, and Code Design, RW Visual Based Programming: Transition, Condition

Complexity: ★★☆

Robo's Story:

Imagine that you need to find something under your bed, where it is very dark. Which device would you use?

Does this device work all the time? Can it be switched off and on? How? With a button, perhaps?

Project Goal:

Now we will transform Robo into a new device with a button; a Robo-flashlight which can be switched on and off.

Day 2: Project 4. Robo is a Flashlight

robo™ wunderkind

Modules:

Main Block

RGB LED

Button

Program:

Conditions

Button

Focus:

- **Robotics:** Button, devices with buttons.
- **Computer Science:** User Input – Parameters of Actions.
- **Robo Code App:** Button condition.

Objectives:

To build a Robo-flashlight and create a simple program that includes some of the Basic Actions and Button condition.

Learning Outcomes:

- I know and can explain what an algorithm is and can use it for programming a Robo-device;
- I know and can explain what button is;
- I can consider how to combine modules to build the Robo-flashlight;
- I know what Transition is and how it happens in both a sequential code and parallel execution;
- I can create a simple program that includes some of the Basic Actions, Connections between them;
- I can modify the code with the Button Condition.

Key Vocabulary:

- Push-button, condition;
- Modules: Button;
- Robo Code App: Transition, Conditions, Button condition.

Day 2: Project 4. Robo is a Flashlight

robo™ wunderkind

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** What is a device? Which devices did we build and program last time and why? What types of code did we program? **Recall** the terminology: **sequential code, parallel execution.**
2. **Analyze:** Tell **Robo's Story**, discuss, and set the **project goal** together with your students.

Guided Activity *15 – 25 min*

3. **Plan:** Decide which **Modules and Coding Actions** you will need for a project and why. Make the step-by-step instruction for Robo-flashlight to perform, introduce the term **"Algorithm."**

   1) Robo-flashlight is switched off – nothing happens. => 2) Button is clicked. => 3) Robo-flashlight emits light.
   4) Robo-flashlight emits light. => 5) Button is clicked. => 6) Robo-flashlight is switched off

4. **Build** a Robo-flashlight and **program** it using the **Algorithm**. Let students discover the right code by themselves guiding them through the steps:
   - Add the **Button** to a build. **Ask:** Which new Module will we need for the Robo-flashlight? What is the button? Which everyday devices have buttons?
   - Learn about **Wait action. Ask:** How can we program. 1)Robo-flashlight is switched off – nothing happens. What does it mean – Robo performs nothing?
   - Add **Visuals** to the code: 3) Robo-flashlight emits light. And **connect Visuals to the Wait action.**
   - Learn about **Transition:** The act of changing from one State to another. Try different **lifespan settings** to see when the Transition happens. Program an **indefinite lifespan** for the **Wait action.** Discuss that the **Transition doesn't happen** in this case.
   - Learn about **Conditions:** A special icon that **makes** the **Transition between** two **Actions happen.** We need to set the Condition if 1) we have a situation in which Transition doesn't happen automatically – the lifespan is infinite, or 2) we only want the Transition to happen in the concrete situation.
   - Explore **Conditions**: Find them in the Action Dock, discuss their design and how they are different from the Actions – they look like stickers. **Ask:** What **kind of Condition** we might need in our situation?
   - Program the **Button condition.** Find the Button condition in the Action dock, and place it on the Connection between two Actions. Discuss that the Condition has to be placed on the Connections to indicate which Transition it regulates. Try it out.
   - **Modify a code** accordingly the Algorithm: 4) same **Constant Light action** + 5) **Connection** back to Wait action + 6) same **Wait action**. Discuss: in order to complete the Algorithm, we modified the code into the Loop; but the second Button condition is missing for the second **Connection**.
   - Program the second **Button condition:** place it on the second Connection in the Loop. Try it out.
5. **Reflect:** Discuss the process of coding and the final solution; pay attention to the Button as a Robo's module, and how the Button condition works.

Independent Activity *30 – 45 min*

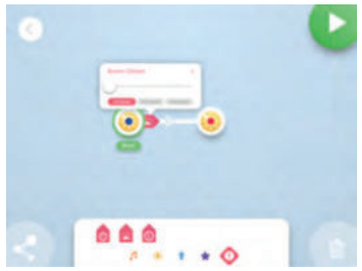
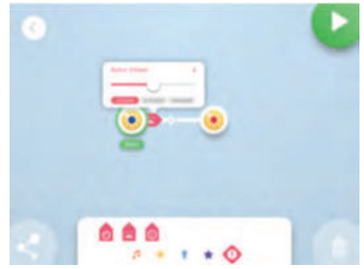
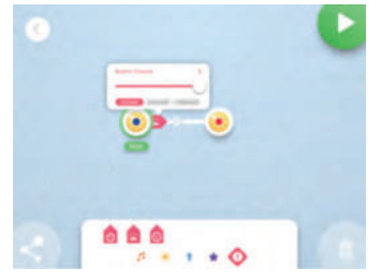
6. **Individual or group activity:**
   - Think about which Robo-device with the Button can be built and how the Button condition can be used in the code for it. Plan, build, and program to carry out the project. Use some materials to customize your Robo.
   - ***Presentation*** *(optional): Students present their projects to the class and give constructive feedback to each other.*

Day 2: Project 4. ✱ Additional Activity

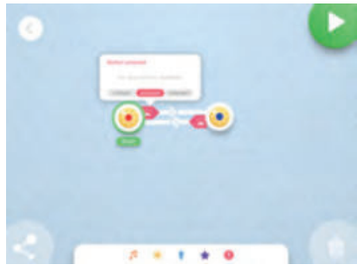
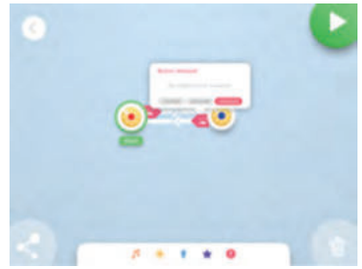
Goal: Explore settings for the **Button condition.**

1. **Ask:** Which devices with buttons do you know? How does a button work? Are all the buttons the same or different? Can we program the Button for Robo differently, too?

2. **Program** the **Button condition** and change the **Settings** for the number of clicks – Transition happens if:
   - The Button is clicked once;
   - The Button is clicked 2 times;
   - The Button is clicked 3 times.

3. Program the **Loop** and add the Button conditions on both Transitions:
   - The first Transition happens if the Button is pressed;
   - The second Transition happens if the Button is released.

4. **Play around** – examples of the tasks:
   - Robo-flashlight makes sounds each time it switches on / off;
   - Robo-flashlight changes the colors if the Button is pressed once; and switches on / off if the button is pressed twice;
   - Robo-flashlight changes the mode – Constant light / Blink actions if the Button is pressed / released.

5. **Sum up:** The function of the Button (module), the function of the Button condition in the code, its settings, and how they influence Robo's performance.

Day 3: Project 5. Robo is a Smart Alarm Clock

robo™ wunderkind

Concepts:
Smart devices, Mechanical, and Code Design, RW Visual Based Programming: Transition, Condition

Complexity: ★★★

Robo's Story:

As you know, our Robo can transform into different devices! Because of this, it can help you with everyday tasks. What exactly is a smart device? Let's answer this question together.

Imagine that you need to get up for a school early in the morning; how can we make it fun? What if you are afraid of oversleeping?

Project Goal:

We will learn about smart devices and transform our Robo in one of them: a Robo-Smart Alarm Clock

Day 3: Project 5. Robo is a Smart Alarm Clock

Modules:


Main Block


RGB LED


DC Motor


Button


Wheel

Program:


Conditions Clock Timer Button

Focus:

- **Robotics:** Outputs – Sounds, RGB Light, DC Motor.
- **Computer Science:** Algorithm.
- **Robo Code App:** Clock, Timer, and Button conditions.

Objectives:

To build a Robo-Smart Alarm Clock and create a simple program that includes some of the Basic Actions and the Clock / Timer and Button conditions.

Learning Outcomes:

- I know what an Algorithm is and can use it to create a code for a Robo-Smart Device;
- I can consider how to combine modules to build the Robo-Alarm Clock;
- I can create a simple program that includes some of the Basic Actions, Connections between them;
- I can modify the code with Clock, Timer, and Button conditions.

Key Vocabulary:

- Algorithm;
- Robo Code App: Timer and Button conditions.

Day 3: Project 5. Robo is a Smart Alarm Clock

robo™ wunderkind

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** Why do people create robots? What is a smart device? How is the smart device connected to other devices? Have you ever used the smart device – if so, which one? Which devices did we build and program last time? Which Conditions did we use in the code designed for it? How did we plan our code? **Recall the term:** We made step-by-step instruction – **Algorithm**.
2. **Analyze:** Tell **Robo's Story**, discuss, and set the **project goal** together with your students.

Guided Activity *15 – 25 min*

3. **Plan:** Decide which **Modules and Coding Actions** you will need for a project and why. Make the step-by-step instruction – **Algorithm** for Robo-flashlight to perform:

   1) Robo-Alarm Clock is waiting – nothing happens => 2) Time to wake up => 3) Robo-Alarm Clock performs an alarm code (infinite)

   4) IF the Button is clicked – 5) Robo-alarm clock is switched off.
4. **Build** a **Robo-Amart Alarm Clock** and **program** it using the **Algorithm**. Let students discover the right code by themselves guiding them through the steps:
   - **Build** a Robo-Smart Alarm Clock and discuss which modules you add to the build and why.
   - **Program** 1) Robo-Alarm Clock is waiting – nothing happens – **Wait action.** Recall how this Action works.
   - **Program** 3) Robo-Alarm Clock performs an alarm code (infinite) – **Sounds, Visuals, Movement** as the sequential logic or parallel execution.
   - **Ask**: How to program step 2) Time to wake up? **Recall**: What is a Condition? How does **Condition** influence the **Transition** between two Actions? How do the Conditions look and work in the **Robo Code App**? Which Condition did we program last time?
   - Learn about the **Timer** and **Clock conditions:** Robo wakes you up at a certain time. Try both Conditions and discuss the difference between them.
   - **Add Button** to the build and program **Button condition:** *4) IF the Button is clicked – 5) Robo-Alarm Clock is switched off.*
   - **Play Around:** Try different settings for all the Actions and Conditions in the code.
5. **Reflect:** Discuss the process of building and coding and the final solution; pay attention to the Timer and Clock conditions and how they work.

Independent Activity *30 – 45 min*

6. **Individual or group activity:**
   - Think about which Robo-Smart Device can be built and how the Button, Timer or Clock conditions can be used in the code for it. Plan, build, and program to carry out the project. Use some materials to customize your Robo.
   - ***Presentation*** *(optional): Students present their projects to the class and give constructive feedback to each other.*

Reflexion & Feedback *8 – 10 min*

7. **Sum up:** Transition, Wait action, Button, Timer or Clock conditions. **Receive feedback:** about the complexity of tasks and activities.
8. **Clean up:** Teach students to take care of the devices they use – RW Modules and the tablets.

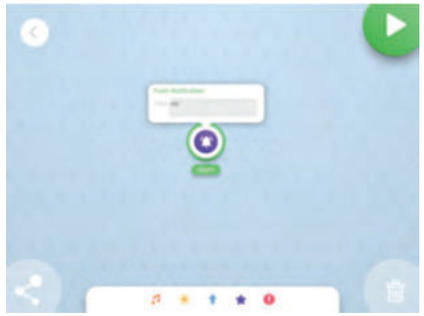
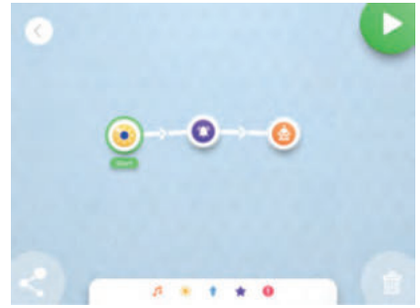
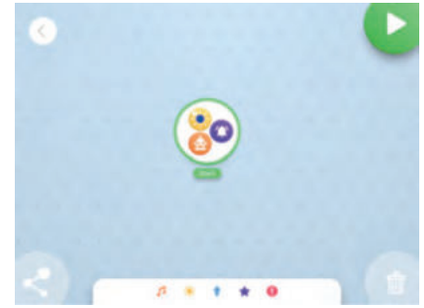
Day 2: Project 4. ✱ Additional Activity

robo™ wunderkind

Goal: Learn about the **Push Notification action.**

1. **Ask:** How else can smart devices notify us? Do you know what the **push notification** is? Have you ever seen push notification(s)? Which devices do send push notifications? Can our Robo send the push notification?

2. Explore the **Push Notification action:**
   - Find the Push Notification action in the **Action Dock** and discuss how it looks: color and icon.
   - Program the **first Push Notification action:** type the text and try it out; discuss what happens on the screen.
   - Program **different Push Notification actions:** change the text.
   - Add **Actions before or after** the Push Notification action and connect them in **sequential code.**
   - Add the **Push Notification action** and another Action in one **State** to form the **Parallel Execution.**

3. **Sum up:**
   - A push notification is a message that pops up on a device such as a tablet or mobile phone. Different Apps can send them at any time; users don't have to be in the app or using their devices to receive them.
   - The Robo Code App can also send the push notification(s): they pop up on the screen when the code is running.

4. **Add the Push Notification action** into the code for **the Robo-Smart Alarm Clock:**
   - Program Robo to send a message when it wakes you up, for example: "Good Morning!"

5. Add the **Push Notification action** in **own project:**
   - Think about which Robo-Smart Device can send the push notification. Plan, build, and program to carry out the project.

Day 3: Project 6. Robo is a Smart Pet!

robo™ wunderkind

Concepts:
Smart devices, Sensor (Input), Mechanical and Code Design, RW Visual Based Programming: Transition, Condition

Complexity: ★★★

Robo's Story:

Some people can't have a pet, due to many different circumstances. In today's modern life, robots can transform into different creatures. Can they become pets?

Project Goal:

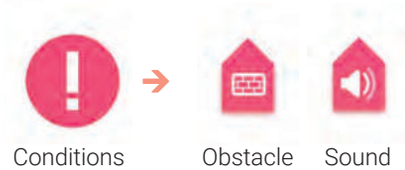
Today we will build and program a Robo-Smart Pet.

Day 3: Project 6. Robo is a Smart Pet!

robo™ wunderkind

Modules:

Program:



Focus:

- **Robotics:** Mechanical design; Outputs: Sounds, RGB Light, DC and Servo Motors.
- **Computer Science:** Algorithm, Sensor Input.
- **Robo Code App:** Obstacle and Sound conditions.

Objectives:

To build a Robo-pet and create a simple program that includes some of the Basic Actions and the Sound and Obstacle conditions.

Learning Outcomes:

- I know and can explain what an algorithm is and can use it for programming a Robo-Smart Device;
- I can consider how to combine modules to build the Robo-pet;
- I know what Transition is and how it happens in both a sequential code and parallel execution;
- I can create a simple program that includes some of the Basic Actions and Connections between them;
- I can modify the code with the Obstacle and Sound conditions.

Key Vocabulary:

- Smart device, sensor (input), distance, decibel;
- Modules: Distance Sensor, Wired Connector;
- Robo Code App: Obstacle and Sound conditions.

Day 3: Project 6. Robo is a Smart Pet!

robo™ wunderkind

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** Which smart device did we build and program last time? Which Conditions did we use in the code designed for it? **Discuss**: Do you have a pet? Do you take care of your pet? Is it difficult? Why don't some people have pets, even if they like them? Can all people adapt to all pets? Can we create a robot-pet? Have you ever seen the robot which looks like a pet?
2. **Analyze:** Tell **Robo's Story**, discuss, and set the **project goal** together with your students.

Guided Activity *15 – 25 min*

3. **Plan and carry out the first project:**
   - Make the Algorithm – step-by-step instruction for Robo-pet to perform.
     1) Robo-pet goes forward => 2) Robo-pet sees the wall => 3) Robo-pet stops
   - Decide which **Modules and Actions** you will need for a project and why; pay attention to the **Wired connector** and its function;
   - Program the steps 1) Robo-pet goes forward – **Drive action** and 3) Robo-pet stops – **Wait action;**
   - Find and explore the **Obstacle condition:** the Robo-pet detects and reacts to the obstacle. Play around and try different settings for it. Introduce the term – **Sensor** and discuss its function.
4. **Plan and carry out the second project:**
   - Make the Algorithm – step-by-step instruction for Robo-pet to perform.
     1) Robo-pet is sleeping => 2) Robo-pet hears the sound => 3) Robo-pet wakes up
   - Decide which **Modules and Actions** you will need for a project and why; pay attention to the **Wired connector** and its function;
   - Program the steps 1) Robo-pet is sleeping – **Wait action** and 3) Robo-pet wakes up – some Basic **Actions** such as **Sounds, Visuals** or **Movement;**
   - Find and explore the **Sound condition:** the Robo-pet reacts to the sound. Play around and try different settings for it. Recall: Sensor and its function.
5. **Reflect:** Discuss the process of coding and the final solution for each project; pay attention to the Obstacle and Sound conditions and how they work.

Independent Activity *30 – 45 min*

6. **Individual or group activity:**
   - Think about what your Robo-smart pet will do and how the Obstacle and Sound conditions can be used in the code for it. Plan, make an Algorithm, build and program to carry out the project. Use some materials to customize your Robo.
   - ***Presentation*** *(optional): Students present their projects to the class and give constructive feedback to each other.*

Reflexion & Feedback *8 – 10 min*

7. **Sum up:** Obstacle and Sound conditions. **Receive feedback:** about the complexity of tasks and activities.
8. **Clean up:** Teach students to take care of the devices they use – RW Modules and the tablets.

Day 2: Project 6. ✱ Additional Activity

robo™ wunderkind

Goal: Explore **Distance and Sound** Sensors and program more advanced settings for the **Obstacle and Sound conditions.**

1. **Ask:** What is a sensor? What are the different types of sensors? How do people use them? In which devices these sensors can be used? Discuss different sensors and their possible use:
   - Temperature Sensor
   - Proximity Sensor
   - Accelerometer
   - Pressure Sensor
   - Light Sensor
   - Ultrasonic Sensor
   - Smoke and Gas Sensor
   - Alcohol Sensor

2. Explore the **Distance Sensor** and program the **Obstacle condition:**
   - Discuss **centimeter** as a measuring unit of distance;
   - **Measure** the distance between different objects using the Control in the **Robo Live App;**
   - Open the **Robo Code App** and discuss the **Less and Greater Than Symbols** in the **Obstacle condition** settings;
   - Program a Robo-device to **indicate** (make a sound, light signal or movement) the certain **distance**; pay attention to Less and Greater Than Symbols;
   - Program a Robo-vehicle to drive until a certain distance before the obstacle, then stop or turn around;
   - Use **more than one Obstacle condition** in a code.

3. Explore the **Sound Sensor** and program the **Sound condition:**
   - Discuss **decibel** as a measuring unit of a sound level;
   - **Measure** the sound level of different sounds in the classroom using the **Control** in the **Robo Live App;**
   - Open the **Robo Code App** and discuss the **Less and Greater Than Symbols** the **Sound condition** settings;
   - Program a Robo-device to **indicate** (make a sound, light signal or movement) the certain **sound level;**
   - Program a Robo-vehicle drive until the sound of a certain sound level is made, then stop or turn around;
   - Use **more than one Sound condition** in a code.

4. Create **own project:** a smart device with either **Distance or Sound sensor:** Plan, build and program to carry out the project.

Day 4: Project 7 – I & II Parts.
Students make their own projects for the final presentation

robo™ wunderkind

Concepts:
Engineering Design Process, Mechanical and Code design

Complexity: ★★★

Robo's Story:

Now it is your turn to decide which device or character you would like your Robo to transform into! Create your own robot using all the Modules & Connectors you want, including additional materials to create an environment; use the Robo Live and Robo Code Apps to control and program your robo. Remember: to build and program the Robo, we also need to come up with a reason as to why we need this particular Robo-device or character, and be able to explain how it will help us.

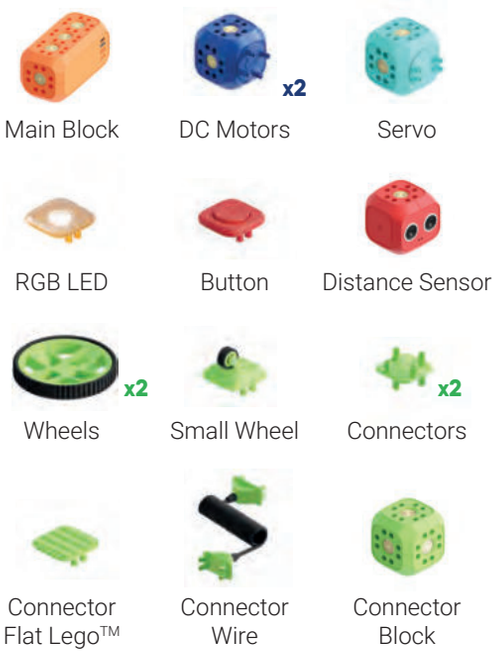
Project Goal:

To transform your Robo into your own project, follow the steps of the Engineering Design Process.

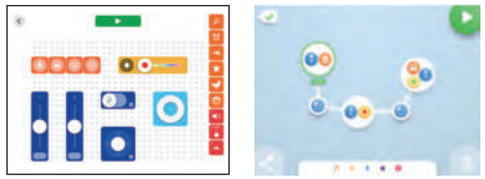
Day 4: Project 7 – I & II Parts.
Students make their own projects for the final presentation

robo™ wunderkind

Modules:



Control and Program:



Focus:

- **Robotics:** Outputs – Sounds, RGB Light, DC and Servo Motors.
- **Computer Science:** User Input regarding the set task; Mechanical and code Design, Engineering Design Process.

Objectives:

To create their own project with RW robotics set – build and program a robot accordingly its function using all the learned Modules and RW Visual Based Programming language.

Learning Outcomes:

- I can come up with a story and reason to build a particular robot;
- I know and can follow the steps of the Engineering Design Process to deal with the set task;
- I can build and program my own Robo-project using all the learned Modules and RW Visual Based Programming language;
- I can create an environment for it using different materials;
- I can present my project to the class and give constructive feedback to other students.

Key Vocabulary:

- Project, Engineering Design Process; Mechanical and code Design.

Additional Materials:

- Printable cards with Engineering Design Process steps;
- *Optional: Lego™ bricks, colored paper, and/or other materials.*

Day 4: Project 7 – Part I

robo™ wunderkind

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** How many projects did we do with Robo? What was the reason for each project? How did we create each project? Have you ever created your own project in your school or daily life? How did you do it? Is this process similar to what we did for Robo? Do you think that there is a cohesive plan for creating a project? Introduce a term: **Engineering Design Process.**
2. **Analyze:** Tell **Robo's Story**, discuss, and set the **project goal** together with your students.

Guided Activity *15 – 25 min*

3. Learn about the **Engineering Design Process.** Ask students to describe how they would make a project; **define each step** together with students, hang the card for this step on the board.
   - **Step 1. Identify a reason: What is the problem or idea?** Define the purpose of particular Robo-project. Reframe it, say it aloud, or write it down; be able to explain the reason.
   - **Step 2. Brainstorm: What are the solutions?** Brainstorm as many solutions as possible, but do not evaluate them. At this step, the solutions don't necessarily need to be good.
   - **Step 3. Evaluate and pick one: What would happen if...?** Think about the pros and cons of each solution, discuss, and rank them. Pick the best solution.
   - **Step 4. Sketch and plan: What will I need?** Make a sketch and decide which Robo's Modules or other materials you will need for the project.
   - **Step 5. Work on a solution: Build and program, test, repeat!** Work on your idea and try it out. If the first solution doesn't work, discuss why and move on to another one. It is important to keep trying until the problem is solved. Do not lose motivation— not all problems are easy to solve!
   - **Step 6. Finalize: Is everything ready?** As soon as you find the best solution for your project, finalize it; create an environment for your Robo or customize it, and check if everything is ready for a presentation.
   - **Step 7. Present the solution.** Show your project to the class, ask for feedback.
   - **Step 8. Reflect: How was it?** As soon as the problem is solved, reflect on the process and ask yourself: What worked? What didn't? What can you do differently next time?

Independent Activity *30 – 45 min*

4. **Practice the Engineering Design Process: Steps 1 – 4.** Students follow the steps of the **Engineering Design Process** on the board in order to **plan** their own Robo-project. They can work independently, in pairs, in small groups, or collaborate on a shared project with two or more robots.

Reflexion & Feedback *8 – 10 min*

5. **Reflect:** Discuss the project ideas and plans together; give constructive feedback to each other in order to improve the project plans.

Day 4: Project 7 – Part II

robo™ wunderkind

Activity Stages:

Lead-in
7 – 10 min

1. **Recall:** The cohesive plan for creating a project – **Engineering Design Process.**
2. **Analyze:** Recall the Robo's Story and the lesson plan.
3. Recall the first 4 Steps of **Engineering Design Process** and ask students to share the **project ideas t**hey have created in Part I:
   - **Step 1.** Identify a reason: What is the problem or idea?
   - **Step 2.** Brainstorm: What are the solutions?
   - **Step 3.** Evaluate and pick one.
   - **Step 4.** Sketch and plan: What will I need?
4. Recall the farther steps of the **Engineering Design Process** and focus on the S**teps 5-6:** hang the card for this step on the board.
   - **Step 5. Work on a solution: Build and program, test, repeat!** Work on your idea and try it out. If the first solution doesn't work, discuss why and move on to another one. It is important to keep trying until the problem is solved. Do not lose motivation! Not all problems are easy to solve.
   - **Step 6. Finalize: Is everything ready?** As soon as you find the best solution for your project, finalize it: create an environment for your Robo or customize it, and check if everything is ready for a presentation.
   - **Step 7.** Present the solution.
   - **Step 8.** Reflect: How was it?

Independent Activity
30 – 45 min

5. **Practice the Engineering Design Process: Steps 5 – 6.** Students follow the steps of the **Engineering Design Process** on the board in order to **work on the solution and finalize** their own Robo-projects. They can work independently, in pairs, in small groups, or collaborate on a shared project with two or more robots. Use different materials to create an environment and customize robots.
   - *Prepare the presentation (Optional): Help the students to take pictures of the Robo, take screenshots of the code, and create the presentation for the final day.*
6. **Presentation to the class:** Students present their projects to the class and give constructive feedback to each other in order to **prepare for the final presentation.**

Reflexion & Feedback
8 – 10 min

7. **Reflect:** discuss the process of coding and the final solution for each project; pay attention to Step 6 – **Finalize: Is everything ready?**

Day 5: Final Presentation

robo™ wunderkind

Concepts:
Engineering Design Process, Mechanical, and Code Design

Complexity: ★★★

Project Goal:

On the very last day of the camp, students will **present their final projects** to their parents. They will share their personal **experiences** during the process of creating and coding, tell stories, show projects, and display **new knowledge.**

Key vocabulary for teachers

robo™ wunderkind

I. Robotics

Robot – A machine capable of carrying out a complex series of actions automatically. Robots can be guided by an external control device – such as a remote controller – or they can be pre-programmed to behave autonomously. Robots are created by people to help with many different tasks which may sometimes be too complex, too dangerous, or simply too repetitive to do ourselves.

Robotics – An interdisciplinary branch of engineering and science utilized in order to deal with the design, construction, operation, and use of robots, as well the control, sensory feedback, and information processing of computer systems.

Engineering – The process of creating and building technological solutions and products by using math and science. **Engineers** solve problems with their inventions. There are several branches of engineering.

Electricity – A type of energy that can build up in one place or flow from one place to another; it is used to make all electrical devices function. **Robo Wunderkind** modules are powered by a battery inside the Main Block. Power is passed from module to module through the **Connectors** (**Pogo-Pins** on them) and **Faces** on each module.

Wired (Local) Communication – A type of electrical communication between electronic devices which transfers information over a wire-based communication technology – connectors. In **Robo Wunderkind robotics kit** there are two types of connectors: Universal Connectors and Wired Connector which transfer the signals – commands from the **Main Block** to other modules.

Wireless Communication – The type of communication which transfers information over a distance without the use of “hard wired” connection (like Pogo-pins on the Robo Wunderkind modules). The distances involved may be short (when using a television remote control) or very long (thousands or even millions of kilometers for radio communications). The **Robo Code and Robo Live Apps** send the information – commands for Robo to perform through wireless communication via Bluetooth.

Bluetooth signals – Is a wireless technology standard for exchanging data over short distances.

Key vocabulary for teachers

robo™ wunderkind

Computer – A device for working with information. The information can be numbers, words, pictures, movies, or sounds. Computer information is also called data. Computers can process huge amounts of data very quickly. They also store and display data. People use computers every day: at work, at school, and at home. Computers are used in factories to control how things are made and in offices to keep records, for example.

There is a small computer inside the **Robo's Main Block**, which sends and receives the signals from the tablet and processes the information so that we can control it. This is why the Main Block always has to be in every project in order for all other modules to work.

Electronic device – A device that achieves its purpose through electrical means. There is a wide range of electronic devices people use every day such as laptops, mobile phones, cameras, fans, ovens, washing machines, game consoles, printers, radios, and Robo, of course!

Smart device – An electronic device, generally connected to other devices or networks (via Wireless Communication) and can operate interactively and autonomously to some extent.

Sensor – A device that receives a physical signal or stimulus – Physical Input (such as sound, pressure or light) and responds to it in a distinctive manner. There are Sound and Distance sensors as well as Button in Robo Wunderkind Education Kit and Motion and Light sensors in Advanced Kit.

Button (Push-button) – Is a simple switch mechanism for controlling some aspect of a machine or a process. In Robo Wunderkind robotics kit there is a red **Button** which can be programmed as **Button condition** in the **Robo Code App**.

Remote control – A method for controlling a machine from a distance by using wireless signals, it happens in real time. For example, the remote control for television when you press buttons to change channels.

Digital literacy – An ability to find, evaluate, compose and create clear information through using various digital platforms. It includes both the **practical software skills** and **critical thinking** which helps to stay safe online. Digital literacy is evaluated by an individual's grammar, composition, typing skills and ability to produce writings, images, audio and designs using technology. In **Robo Wunderkind Curriculum** the **numerous skills** related to the digital literacy are covered such as rules for **safe use of devices, consuming information through digital media** and, most importantly, **ability to produce technology** through coding.

Key vocabulary for teachers

robo™ wunderkind

Problem-solving – Is the act of finding a solution for a problem. The steps for the problem solving are slightly different depending on the discipline and strategy but it always involves defining a problem, identifying, prioritizing and selecting alternatives for a solution, implementing a solution, and reflection on this solution.

In **Robo Wunderking Curriculum** students go through **4 steps** in order to solve a problem using Robo Wunderkind robotics kit and Robo Code or Robo Live App:
1) Identify a problem => 2) Plan the solution(s) => 3) Work on the solution: build and program => 4) Reflect

---

Engineering Design Process – A series of steps that engineers follow to come up with a solution to a problem. It involves cognitive, strategic and practical processes by which design concepts are developed.

In **Robo Wunderkind Curriculum** students learn about and go through 8 stages of **Engineering Design Process** in order to create their own Robo-project.

Design – Is the intentional creation of a plan in order to fashion, execute, or construct a certain object.
In **Robo Wunderkind Curriculum** we approach 3 types of design:

1) Mechanical Design – Robo Modules
- How to design a robot efficiently, so it performs the set task, doesn't crash or have accidents?
- How to make an efficient build – attach only those Modules which you need for your goals?
- How do different builds and configurations provide the same or different results?

2) Control Design – Robo Live App
- You have all the controls you need – not less;
- You have only those controls on the screen which you need (not more);
- You organize the Controls on the screen so it's convenient to use them in the Play modeю

3) Code Design – Robo Code App
- Your code consists of all the Actions, States, Connections you need to achieve the project's goal(s) – not less;
- Your code consists of all the Actions, States, Connections you need to achieve the project's goal(s) – not more;
- Your code is clear and easy to read by others.

Key vocabulary for teachers

robo™ wunderkind

II. Computer Science

Algorithm –	Step-by-step solution of one task; each step is a clear instruction. A simple example of an algorithm is a cooking recipe, where you have one by one instruction in order to cook one final dish.
Program code –	A set of instructions which tells a computer what to do; a sequence of short commands, one after another.
Programming language	In order for you to communicate with a computer (and to get it to execute your instructions), you must speak its language. There are a number of different programming languages, some are very complicated while others are similar to spoken English. In the **Robo Code App**, we use a special visual coding language. These 3 terms **(Algorithm, program, programming language)** are interconnected. To help students understand them better, we can say that: • When we have one complex task, we can break up it into a set of smaller, individual instructions – create an **algorithm**; • We can use a **programming language** to write those instructions in the language computer understands – thus, we create a **code**.
Robo Wunderkind Visual Based Programming –	A unique and intuitive programming interface designed for young children to build State-Machine Based Programs
State-Machine Based Programming –	A State performs Action(s) and Transition to another State based on events. • **State** – set of Actions which can consist of one or more Actions; • **Connection** – tells the possible Transition to other States (it may happen / may not); • **Transition** – the act of changing from one State to another (= the act of happening); • **Condition** – an icon which compares two numbers and determines the results be true or false, and tells if true – Transition happens or false – nothing happens; • **Action** – an icon which performs an output task; and has different parameters: Lifespan, Speed, Distance etc.

Key vocabulary for teachers

robo™ wunderkind

Sequential logic – A sequence of States where one State leads to only one other State until a program is complete.

Loop – A sequence of States that lead one State to the next that does not terminate the program but the transition back to the starting State (repeats N-times).

Parallel Execution – The execution of several Actions at the same time within a State.

User Input – Is is data provided by the user to the device. It can be digital input such as text to display on a screen or it could be physical like a button click by the user or a key board press on a keyboard.

In the Robo Live App – using the appropriate Control to control the appropriate Module. **In the Robo Code App:**

1) Software Input

- Parameters of Actions such as Action lifespan, brightness, speed, angle or distance;
- Connections drawn between Actions;
- Conditions and Conditions parameters.

2) Sensor Input – data received from Sensors such as sound level, distance before an obstacle or pressure on the Button.

User Output – For Robo Wunderkind Modules the Outputs would be the answers for the questions:

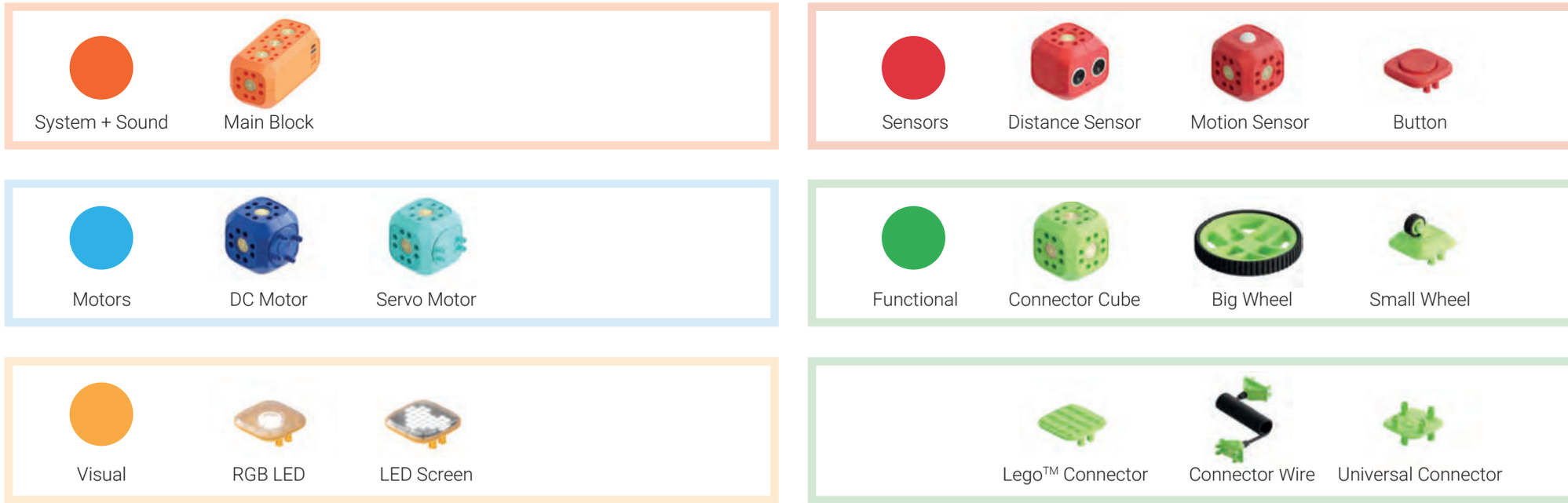
What does the Module do?

- How are Modules different from each other?
- How do they work together?
- How to combine the Modules to achieve the project's goal(s)?

Robo Wunderkind Robotics kit:

robo™ wunderkind

RW Modules' Overview:



Main Block – The "brain" of Robo: contains an embedded computer, battery, and speaker outlet inside.

Connectors – To make all the modules work, you need the electricity and the code signals from the Main Block, which passes through the small **Pogo-pins on Connectors to Modules' faces.**

Pogo-pins – Small metallic pins on the Connectors which help to establish a connection between two RW Modules.

DC Motors rotate around themselves, so you can control or program a robot to drive around;

Servo Motor turns to the **exact angle** from the **zero position.** When you attach the Servo Motor to the other module, you **set the zero-position** for it.

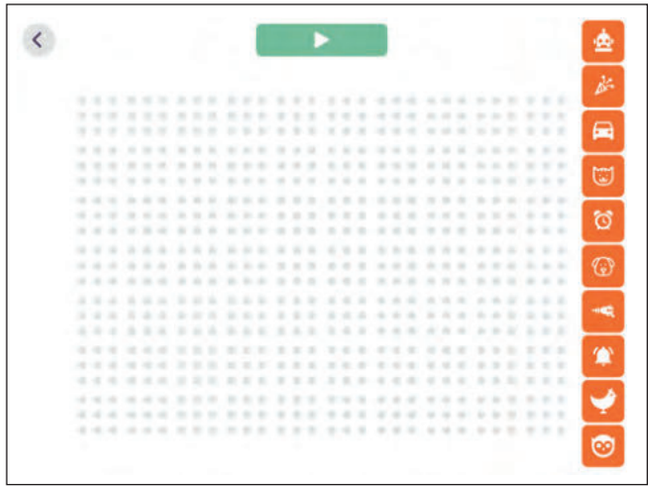
RGB LED – Light – Program or control different light signals.

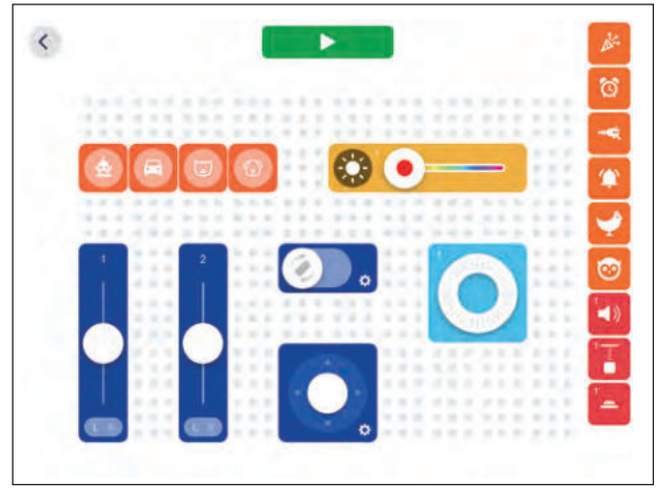
Distance Sensor – There are Distance and Sound Sensors in this Module; Distance sensor detects the distance before an obstacle; and **Sound Sensor** detects the loudness of the sound.

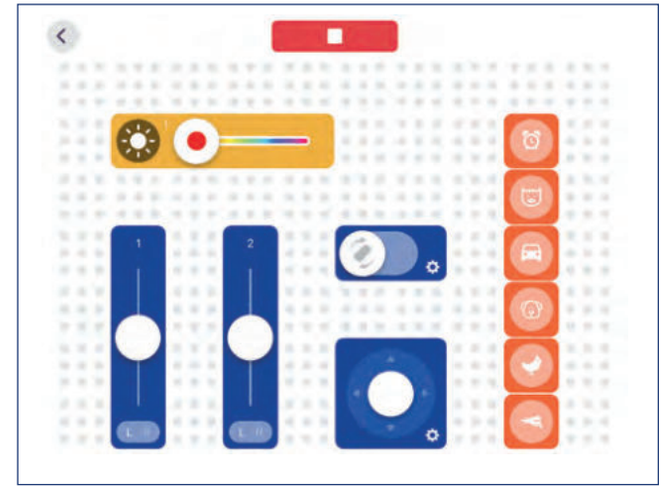
Robo Live App

robo™ wunderkind

① ② ③







① Controlling screen

Here you will see the Controls added.

Controls Menu: See all the Controls available at the moment, drag and drop a Control to the Controlling screen to try it out.

Play / Edit button: When you are ready to play, press the Play button and control your Robo; press Edit button to go back to the editing mode.

Save button: Click it to return to the project screen; don't worry, your project will be saved automatically.

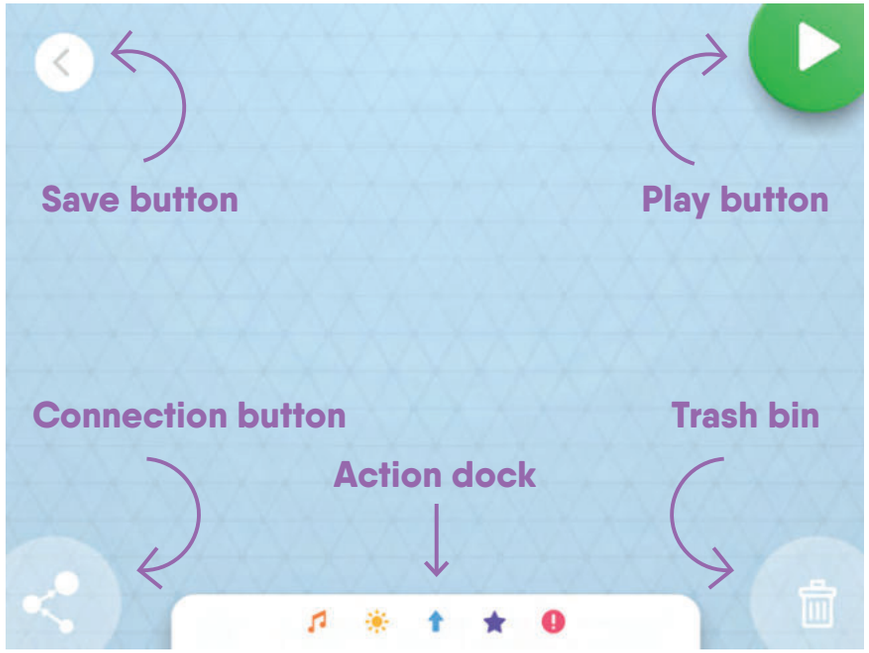
② Controls

- When you create a new project, the Controlling screen is empty;
- Using only the Main Block, you can already add Sound control(s); simply drag and drop them onto the screen;
- When you attach the Module to the Main Block, the Control(s) available for this module appears in Controls Menu automatically;
- You can delete the Control from the Controlling screen if needed.

③ Play Mode

When you are ready to play, press the Play button and control your Robo; press Edit button to go back to the editing mode.

Robo Code App



Programming screen

The unlimited space for your future program.

Actions dock: Here you will find all the Actions and Conditions to program.

Connection button: Use it to connect the Actions into the code.

Trash bin: Use it to delete the Actions or Transitions / Connections.

Play button: Try out your program!

Stop button: When you run your code, the 'Play' button changes to the 'Stop' button; you can stop your Robo any time you want.

Save button: Click it to return to the project screen; don't worry, your project will be saved automatically.

Modules, coding buttons, and icons to print

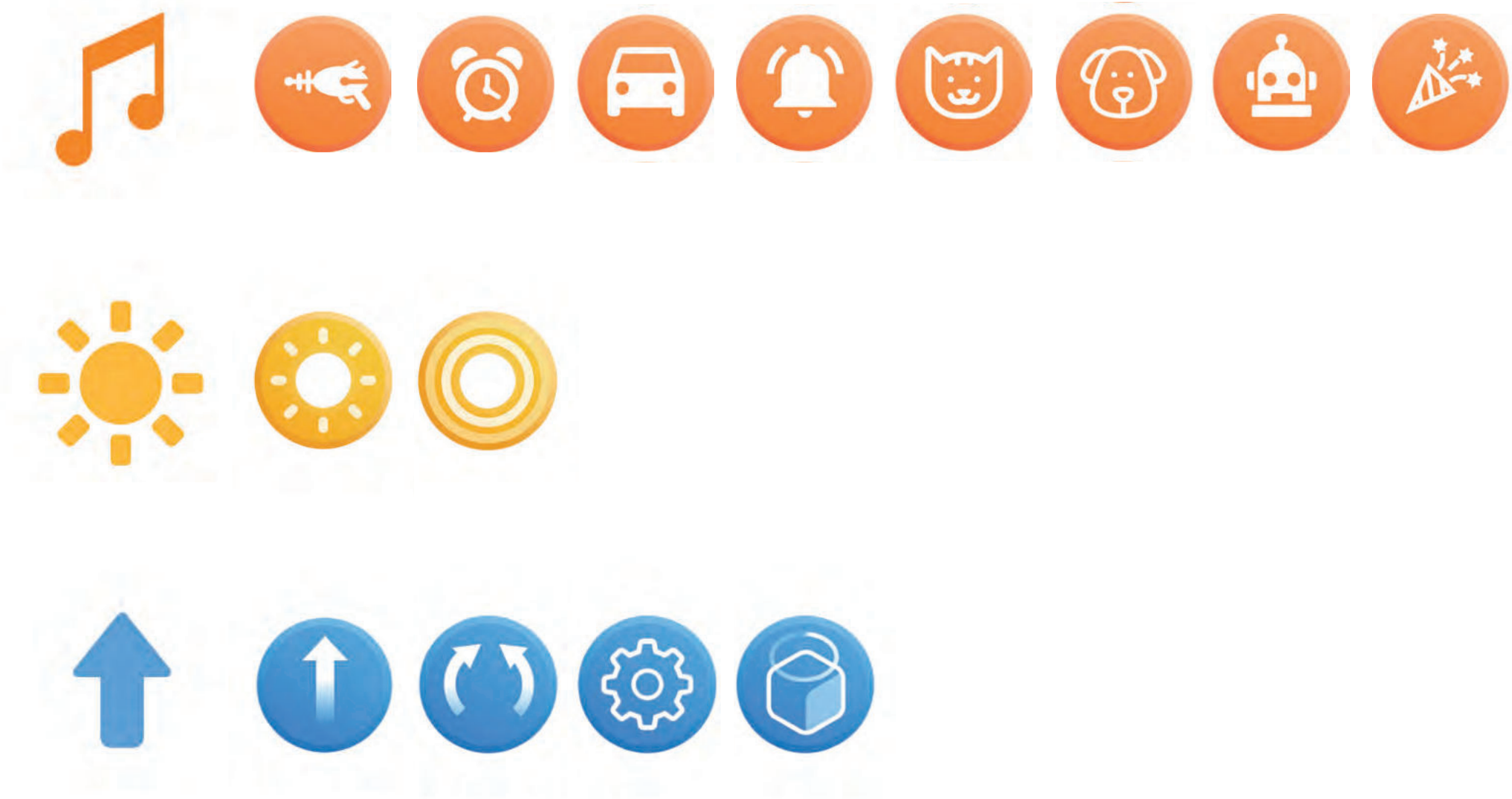
robo™ wunderkind

Modules, coding buttons, and icons to print

robo™ wunderkind

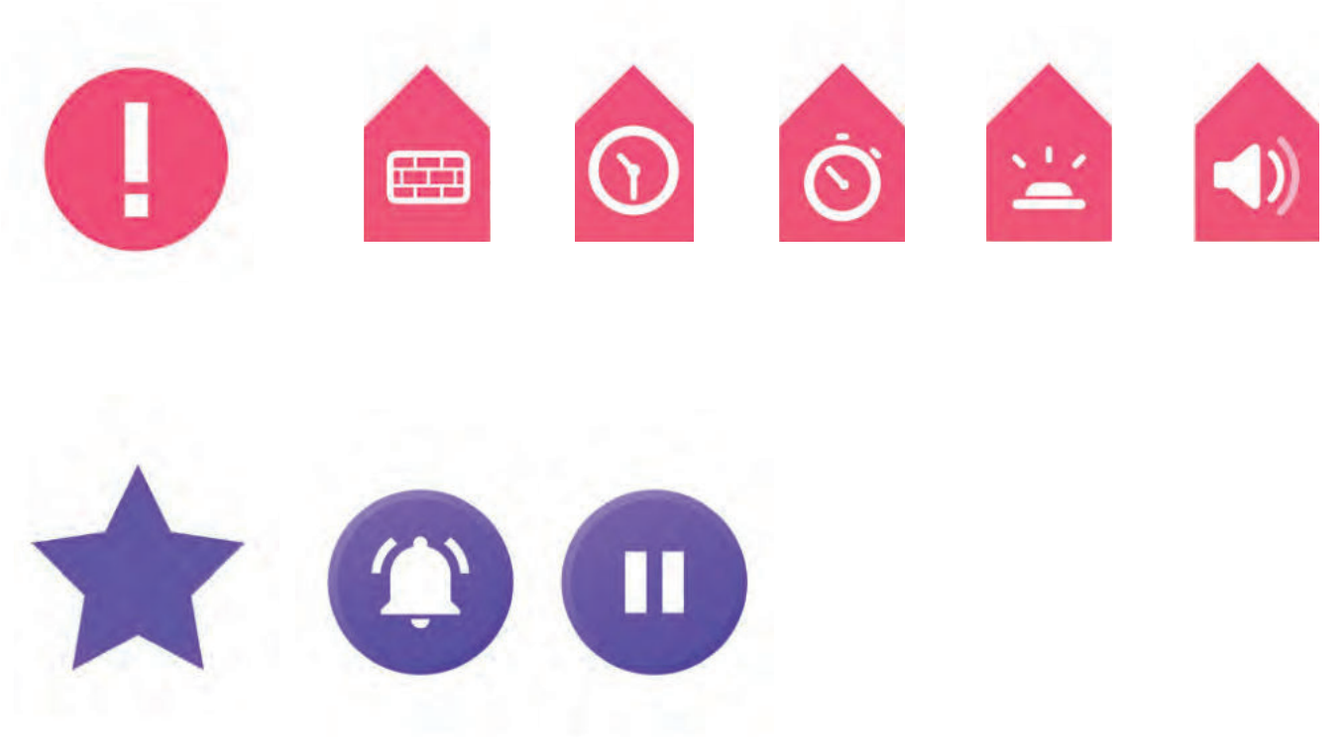
Modules, coding buttons, and icons to print

robo™ wunderkind



Modules, coding buttons, and icons to print

robo™ wunderkind



Cards with the steps of the Engineering Design Process

robo™ wunderkind

Step 1.
Identify a reason: What is the problem or idea?

Step 2. Brainstorm: What are the solutions?

Step 3. Evaluate and pick one.

Step 4. Sketch and plan: What will I need?

Step 5.
Work on a solution: Build and program, test, repeat!

Step 6. Finalize: Is everything ready?

Step 7. Present the solution.

Step 8. Reflect: How was it?