robo™ wunderkind

Curriculum

Block #2: Introduction to Robotics and Computer Science.
Theme topic: Smart Devices

Welcome to the Robo Wunderkind Curriculum!

We are happy to introduce you to the second block of lessons from Robo Wunderkind robotics and coding program. **Block #2 – Smart Devices** consists of **10 lessons** which will introduce your students to the topic of robots' use for **Smart Device Environment** and will bring them to the next level of robotics and computer science experiences such as the purpose of **sensors** in building robots or use of **Conditions** in programming. Our ready-to-use curriculum is made to fully support you during the preparation and entire course of teaching with RW. 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 lessons;
- **Concepts Overview** as well as formulated **Learning Outcomes**;
- **10 Lesson Plans** – well-constructed and easy to follow, with **Additional Activities** that allow you to adjust the complexity level of lessons to specific student needs;
- **Supporting materials** such as **teacher slides, challenge cards, key vocabulary,** and **printable modules' images** to ensure a comfortable teaching experience.

Table of contents:

Key Information

Topic: STEAM subjects **Grades:** 1-4 **Group size:** 6-12 students

All the lessons in Block #2 are **story-based** and linked by one topic: **Smart Devices.** In this series, Robo will transform into various robots / Smart Devices in order to solve different tasks. Students will need to use their **previous knowledge** of **Robo Wunderkind modules**, Visual Based Programming language of **Robo Code App** as well as **Engineering Design Process** in order to analyze, plan, and carry out the projects.

Complexity: The lessons are more complex than the Block #1 and required the previous knowledge of RW robotics kit and Robo Code App. Each lesson includes the basic level as well as a possible modification / additional activity. This opens the possibility of adapting the complexity level of the lessons to the specific needs of your students.

Recommended Prior Knowledge: Students will need prior knowledge about some modules of Robo Wunderkind robotics kit and Visual Based Programming language of Robo Code App, as well as the Engineering Design Process.

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: teacher slides, challenge cards, printable modules' and coding actions' images.

Robotics and Computer Science Concepts Covered in Block #2

robo™ wunderkind

Concepts	Lesson 1	Lesson 2	Lesson 3	Lesson 4	Lesson 5	Lesson 6	Lesson 7	Lesson 8	Lesson 9	Lesson 10
Robotics										
1. Robotics, Engineering	+	+	+	+	+	+	+	+	+	+
2. Electricity:										
• Electrical Power	+	+	+	+	+	+	+	+	+	+
• Local Communication	+	+	+	+	+	+	+	+	+	+
3. Wireless communication	+	+	+	+	+	+	+	+	+	+
4. Design Thinking Process:										
• Mechanical Design	+	+	+	+	+	+	+	+	+	+
• Code Design		+	+	+	+	+	+	+	+	+
Computer Science										
Programming, Code	+	+	+	+	+	+	+	+	+	+
1. State-Machine Based Programming:										
1. Transition	+	+	+	+	+	+	+	+	+	+
2. Condition	+	+	+	+	+	+	+	+	+	+
3. Event					+	+	+	+	+	
2. User Input: Parameters	+	+	+	+	+	+	+	+	+	+
3. Digital literacy	+	+	+	+	+	+	+	+	+	+
4. Algorithm	+	+	+			+		+		
5. Decision Tree				+	+		+		+	+
6. Engineering Design Process	+	+	+	+	+	+	+	+	+	+

Projects' Overview

Projects	Concepts	Complexity	Page
1. Robo Is a Smart Reminder	Smart device, Electricity & Wireless communication, Transition, Condition	☆☆	6
2. Robo Is a Flashlight	Button, Transition, Condition	☆☆	9
3. Robo Is a Smart Lamp	Button, Transition, Condition, Mechanical, and Code Design	☆☆	12
4. Robo Is a Smart Kitchen Timer	Decision Tree, Condition, Mechanical, and Code Design	☆☆	15
5. Robo Is a Smart Alarm Clock	Decision Tree, Condition, Mechanical, and Code Design	☆☆☆	18
6. Robo Is a Smart Vacuum Cleaner	Sensor, Condition, Mechanical, and Code Design	☆☆	21
7. Robo Is a Measuring Device	Decision Tree, Condition, Comparison, Mechanical, and Code Design	☆☆☆	23
8. Robo Is a Sound Detector	Sound level, Sensor, Condition, Mechanical, and Code Design	☆☆	25
9. Robo Keeps Order	Sound level, Decision Tree, Condition, Mechanical, and Code Design	☆☆	27
10. Robo Is a Smart Pet	Artificial intelligence, Decision Tree, Condition, Mechanical and Code Design	☆☆☆	29

Project 1: Robo Is a Smart Reminder

Concepts: Robotics, Electricity & Wireless communication, Transition, Condition, Mechanical, and Code Design

Complexity: ★★☆

Robo's Story:

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

Imagine that you need to be reminded of something important during the day; What device would you use? Can we build it from RW Modules?

Problem Situation:

We need to be reminded of an important event.

Solution:

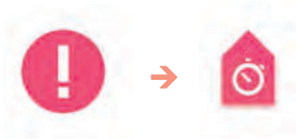
Build and program a Robo-smart reminder.

Project 1: Robo Is a Smart Reminder

Modules:

Main Block

Program:



Conditions Timer

Focus:

- **Robotics:** Smart Devices and their function in everyday life;
- **Robo Code App:** Timer condition.

Objectives:

- To build a Robo smart reminder and create a simple program that includes some of the basic Actions and Timer condition.

Learning Outcomes:

- I know and can explain what a smart device is;
- I can combine modules to build the Robo-smart reminder;
- 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 and modifies it with the Timer condition.

Key vocabulary:

- The device, smart device, (Bluetooth) signals, condition;
- Robo Code App: Transition, Conditions, Timer condition.

Project 1: Robo Is a Smart Reminder

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** What is a device? What is a smart device? How is the smart device connected to other devices? Discuss the term: **Bluetooth signals.**
2. **Analyze:** Tell Robo's Story, identify the **problem situation** and come up with a **theoretical solution.**

Guided Activity *15 – 20 min*

3. **Plan:** decide which Modules and coding Actions you will need for a project and why. Make an **Algorithm**, plan for Robo-reminder to perform: *1) Robo-reminder is waiting for a certain amount of time. => 2) Time is over. => 3) Robo-reminder makes a reminder sound.*
4. **Build** a Robo-smart reminder and **program** it using the **Algorithm**. Let students discover the right code by themselves guiding them through the steps:
   - Create a **code** according to the Algorithm: **1) Wait action** + 2) Wait action's lifespan settings + **3) Sound action.** Discuss the **Wait action's lifespan:** it lasts 10 seconds – it is possible to wait only 10 seconds.
   - Learn about **Transition:** the act of changing from one State to another. Try different lifespan's 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 Actions happen.** We need to set the Condition if 1) we have a situation in which the Transition doesn't happen automatically – the lifespan is infinite, or 2) we want the Transition happens only 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? We need to set a time: **Timer condition.**
   - Program a **Timer condition:** place the Timer condition on the Connection between Actions. Discuss that the Condition has to be placed on the Connections: to indicate which Transition it regulates.
   - **Play around:** Program different codes for Robo-smart reminder with the Timer condition(s) to regulate the time.
5. **Sum up** what a Transition and Condition is, why we might need them in the form of code.

(*) ***Additional activity:*** *the Robo-smart reminder:* ***Mechanical and Code Design.*** *Add RGB Light, DC Motor or Servo motor to the build and program different reminder codes for Robo-smart reminder to perform.*

Independent Activity *15 – 20 min*

6. **Make your own project:** think about which Robo-smart device with the Timer condition also can be built. Plan, build and program to carry out the project. Use some materials to customize your Robo.

Reflexion & Feedback *5 – 7 min*

7. **Sum up:** Transition, Conditions, Timer condition. **Receive feedback** about the complexity of tasks and activities.
8. **Clean up.**

Project 2: Robo Is a Flashlight

Concepts: Button, Electricity & Wireless communication, Transition, Condition, Mechanical, and Code Design.

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? For that reason, it has a button.

Problem Situation:

We need to witch on and off a flashlight to light up a dark place.

Solution:

Build and program a Robo-flashlight with the button to turn it on and off.

Project 2: Robo Is a Flashlight

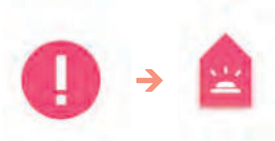
Modules:


Main Block


RGB LED


Button

Program:


Conditions Button

Focus:

- **Robotics:** Button, devices with buttons;
- **Robo Code App:** Button condition.

Objectives:

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

Learning Outcomes:

- 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 and modifies it with the Button condition.

Key vocabulary:

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

Project 2: Robo Is a Flashlight

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** What is a smart device? Which smart device did we build and program last time and why? What is a Transition? What are Conditions and why do we need them in a code? How do Conditions look like in the Robo Code App? **Recall** the terminology: **Bluetooth signals, Transition, Condition.**
2. **Analyze:** Tell Robo's Story, identify the **problem situation** and come up with a **theoretical solution.**

Guided Activity *15 – 20 min*

3. **Plan:** Decide which Modules and coding Actions you will need for a project and why. Make an **Algorithm: plan** for Robo-flashlight to perform:
   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:
   - Learn about the **Button. Ask:** Which new Module will we need for the Robo-flashlight? What is the button? Which everyday devices do have buttons? How does a button work?
   - Create a **code** accordingly the Algorithm: **1) Wait action + 2) Connection + 3) Constant Light action.** Discuss that a Condition is missing between these Actions.
   - 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 a 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. **Sum up** What a button is and does as a Robo's module, how the Button condition works.

(*) ***Additional activity:*** *Explore more about the* ***Button condition.*** *Set different setting: button is clicked once or twice; pressed or realised.*

Independent Activity *15 – 20 min*

6. **Make your own project:** Think about which Robo-smart 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.

Reflexion & Feedback *5 – 7 min*

7. **Sum up:** Transition, Conditions, Button (module), and Button condition. **Receive feedback** about the complexity of tasks and activities.
8. **Clean up.**

Project 3: Robo Is a Smart Lamp

Concepts: Button, Electricity & Wireless communication, Transition, Condition, Mechanical, and Code Design

Complexity: ★★☆

Robo's Story:

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 would you use?

Would be comfortable to switch light's color with a button all the time? Can a smart device do it automatically?

Problem Situation:

We need to color up the room with different lights which change automatically.

Solution:

Build and program a Robo-smart lamp which can be switched on and off; design it so that the color will automatically change at particular times.

Project 3: Robo Is a Smart Lamp

Modules:


Main Block


RGB LED


Button


Wheel


Connector Block

Program:


Conditions → Timer → Button

Focus:

- **Robotics:** Smart lamp bulb;
- **Robo Code App:** Timer and Button conditions.

Objectives:

- To build a Robo-smart lamp and create a simple program that includes some of the basic Actions and the Timer and Button conditions.

Learning Outcomes:

- I know and can explain what a smart lamp bulb is;
- I can consider how to combine modules to build the Robo-smart lamp;
- I can create a simple program that includes some of the basic Actions Connections between and modifies it with the Timer and Button conditions.

Key vocabulary:

- Smart lamp bulb, button, condition;
- Robo Code App: Transition, Conditions, Timer condition.

Project 3: Robo Is a Smart Lamp

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** Which Smart Devices did we build and program last times and why? Which Conditions did we use? What are Conditions and how do they influence the Transitions between the Actions? **Recall** the terminology: **Transition, Condition, Button.**
2. **Analyze:** Tell Robo's Story, identify the **problem situation** and come up with a **theoretical solution.**

Guided Activity *15 – 20 min*

3. **Plan:** Decide which Modules and coding Actions you will need for a project and why. Make an **Algorithm: plan** for Robo-smart lamp to perform: *1) Robo-lamp is switched off – nothing happens. => 2) Button is clicked. => 3) Robo-lamp emits first light. => 4) Robo-lamp switches the light color. => 5) Robo-flashlight emits 2-3 lights in a specific order. => 6) Button is clicked. => 7) Robo-lamp is switched off.*
4. **Build** Robo-smart lamp and **program** it using the **Algorithm.** Let students discover the right code by themselves guiding them through the steps:
   - Create a code accordingly the Algorithm: **1) Wait action + 2) Connection + 3) Constant Light action.** Discuss that a Condition is missing between these Actions and add the **Button condition on the Transition.**
   - **Modify a code** accordingly the Algorithm: **4) Constant Light action with a different setting.** Discuss how we can regulate the time in which Robo changes the lights: use **Action's settings** or the **Timer condition.**
   - **Modify a code** accordingly the Algorithm: **5) Constant Light action** + **6) Connection** back to Wait action + 6) same **Wait action.** Discuss: order to complete the Algorithm, we modified a code into the Loop; but the second Button condition is missing for the second Connection. Program the second **Button condition.**
   - **Play around:** program different chains of Constant Light or Blink actions for Robo-smart lamp to perform. Use both **Action's settings** or the **Timer condition** to regulate the time and the Button condition to switch it on and off.
5. **Sum up** The function of Conditions in a code. What are the advantages and difficulties of using two different types of Conditions in one code?

(*) ***Additional activity:*** *Discuss the specific* ***mechanical and code design*** *for the Robo-smart lamp. Ask: How can we make the lamp more convenient to use? How would you make the code more clear and sufficient?*

Independent Activity *15 – 20 min*

6. **Make your own project:** think about which Robo-smart device which needs the Button and Timer conditions in a code. Plan, build and program to carry out the project. Use some materials to customize your Robo.

Reflexion & Feedback *5 – 7 min*

7. **Sum up:** Transition, Conditions, Button (module), and Button condition. **Receive feedback** about the complexity of tasks and activities.
8. **Clean up.**

Project 4: Robo Is a Smart Kitchen Timer

Concepts: Decision Tree, Transition, Condition, Mechanical, and Code Design

Complexity: ★★☆

Robo's Story:

Imagine that you are cooking a cake and you need to put it in an oven for an exact time. Which smart Robo-device can you help you measure the time and remind you that it's over? Can we set the alarm once again?

Problem Situation:

You want to stop a kitchen timer or have it repeat the set time.

Solution:

Build and program a Robo-smart kitchen timer which can be stopped and can occur repetitively.

Project 4: Robo Is a Smart Kitchen Timer

Modules:


Main Block


RGB LED


Button


Wheel

Program:


Conditions Timer Button

Focus:

- **Computer Science:** Decision tree;
- **Robo Code App:** Timer and Button conditions.

Objectives:

- To build a Robo-smart kitchen timer and use a decision tree to create a simple program that includes some of the basic Actions and the Timer and Button conditions.

Learning Outcomes:

- I know and can explain what a decision tree is and can use it for programming a Robo-device;
- I can consider how to combine modules to build the Robo-kitchen timer;
- I can create a simple program that includes some of the basic Actions, Connections between them; I can modify it with the Timer and Button conditions.

Key vocabulary:

- Decision tree, options;
- Robo Code App: Timer and Button conditions.

Project 4: Robo Is a Smart Kitchen Timer

Activity Stages:

Lead-in *7 – 10 min*

1. **Ask:** Which Smart Devices did we build and program last times and why? Which Conditions did we use? How did we plan our project? Recall the term: **Algorithm.**
2. **Analyze:** Tell Robo's Story, identify the **problem situation** and come up with a **theoretical solution.**

Guided Activity *15 – 20 min*

3. Introduce the term: **Decision Tree.** Discuss how it is different from the previous algorithms. **Plan** the project together with the students **using the Decision Tree.**
   1) Robo-kitchen timer is counting the time – nothing happens. => 2) Time is over => 3) Robo-kitchen timer performs an alarm code ***Decision Tree:*** *=> 4) IF the Button is pressed ONCE – 5) Robo-kitchen timer is counting the time once again => 4) IF the Button is clicked TWICE => 5) Robo-kitchen timer is switched off.*
4. **Build** Robo-kitchen timer and **program** it using the **Decision Tree.** Let students discover the right code by themselves guiding them through the steps:
   - Create a first part of a **code: 1) Wait action** + **2) Connection** + **3) Alarm code** + add the **Timer condition;**
   - **Modify a code – variation #1: 4) Connection** back to Wait action – **Loop + 5) Button condition** with a setting clicked once
   - **Modify a code – variation #2: 4) Add the second Wait action** + 5) Draw the **Connection** from alarm code and add the **Button condition** with a setting **clicked twice.** Discuss the code design with a decision tree: how it is different from the algorithmic code, advantages and disadvantages of its use.
   - **Play around:** program different variations of a decision tree for Robo-smart kitchen timer to perform.
5. **Sum up** Decision Tree; why, how, and what we need to properly use to plan a code.

(*) ***Additional activity:*** *Explore more about the Button condition. Use setting: pressed for variation #1 and released for variation #2 of a decision tree.*

Independent Activity *15 – 20 min*

6. **Make your own project:** think about which Robo-smart device which needs the Button and Timer conditions and Decision Tree to design a code. Plan, build and program to carry out the project. Use some materials to customize your Robo.

Reflexion & Feedback *5 – 7 min*

7. **Sum up:** Transition, Timer and Button conditions. **Receive feedback** about the complexity of tasks and activities.
8. **Clean up.**

Project 5: Robo Is a Smart Alarm Clock

Concepts: Decision Tree, Transition, Condition, Mechanical, and Code Design

Complexity: ★★★

Robo's Story:

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 to oversleep?

Problem Situation:

You need an alarm clock which can repeat in case you don't wake up the first time.

Solution:

Build and program a Robo-smart alarm clock which can repeat if you don't wake up the first time.

Project 5: Robo Is a Smart Alarm Clock

Modules:


Main Block


RGB LED


Button


DC Motors


Wheel

Program:


Conditions Timer Clock Button

Focus:

- **Computer Science:** Decision tree;
- **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 Timer condition.

Learning Outcomes:

- I know what a Decision Tree 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 it with Clock, Timer, and Button conditions.

Key vocabulary:

- Decision tree, options;
- Robo Code App: Timer and Button conditions.

Project 5: Robo Is a Smart Alarm Clock

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? How was it different from the all the codes we had programmed before? Recall the term: **Algorithm, Decision Tree.**
2. **Analyze:** Tell Robo's Story, identify the **problem situation** and come up with a **theoretical solution.**

Guided Activity *15 – 20 min*

3. **Plan:** Decide which Modules and coding Actions you will need for a project and why. Make a **Decision Tree** for Robo-smart alarm clock to perform:
   1) Robo-alarm clock is waiting – nothing happens => 2) Time to wake up => 3) Robo-alarm clock performs an alarm code => ***Decision Tree:*** *=> 4) IF the Button is pressed TWICE – 5) Robo-alarm clock is switched off. => 4) IF the Button is clicked ONCE => 5) Robo-alarm clock repeats an alarm code again in 5 minutes.*
4. **Build** a Robo-alarm clock and **program** it using the **Decision Tree.** Let students discover the right code by themselves guiding them through the steps:
   - Create a first part of a code: **1) Wait action + 2) Connection + 3) Alarm code** + add the Clock condition on the Connection.
   - **Modify a code – variation #1: 4) Connection** back to Wait action – **Loop + 5) Button condition** with a setting **clicked twice.**
   - **Modify a code – variation #2: 4) Add** the second **Wait action** and draw the **Connections** to this Action to form a Loop;
     5) Add the **Button condition** on the Connection to the Wait action and the Timer condition with a setting clicked once. Discuss the **code design** with a Decision Tree: how it is different from the algorithmic code, advantages and disadvantages of its use.
   - **Play around:** program different variations of a decision tree for Robo-smart alarm clock to perform.
5. **Sum up** Decision Tree; how and what we need to use to properly plan a code, the functions of the Timer and Clock conditions and how they are different.

(*) ***Additional activity:*** *Discuss the difference between the Timer and Clock. Approach concept of time using a Robo-smart device and the Timer and Clock conditions.*

Independent Activity *15 – 20 min*

6. **Make your own project:** Think about which Robo-smart device which needs the Timer, Clock and Button conditions and Decision Tree to design a code. Plan, build and program to carry out the project. Use some materials to customize your Robo.

Reflexion & Feedback *5 – 7 min*

7. **Sum up:** Transition, Timer and Button conditions. **Receive feedback** about the complexity of tasks and activities.
8. **Clean up.**

Project 6: Robo Is a Smart Vacuum Cleaner

Concepts: Sensor, Condition, Mechanical, and Code Design

Complexity: ★★☆

Robo's Story:

Imagine that you have to clean your room. Which device will help you? Our Robo can also transform into a smart device, so let's build and program our Robo-smart vacuum cleaner!

Problem Situation:

You need an alarm clock which repeats if you don't wake up the first time.

Solution:

Build and program a Robo-smart alarm clock which can repeat in the case that you don't wake up the first time.

Project 6: Robo Is a Smart Vacuum Cleaner

Modules:


x2
Main Block


Button


DC Motors


x2
Wheel

Program:


Conditions


Obstacle

Focus:

- **Robotics:** Sensors and their functions, distance and Distance sensor;
- **Robo Code App:** Obstacle condition.

Objectives:

- To build a Robo-smart lamp and create a simple program that includes some of the basic Actions and the Obstacle condition.

Learning Outcomes:

- I know and can explain what a sensor is and how it works;
- I know what centimeter is as a measuring unit of distance;
- I can consider how to combine modules to build the Robo-smart vacuum cleaner;
- I can create a simple program that includes some of the basic Actions, Connections between them; I can modify it with the Obstacle condition.

Key vocabulary:

- Distance, centimeter, sensor;
- Robo Code App: Obstacle condition.

Project 7: Robo Is a Measuring Device

Concepts: Decision Tree, Condition, Comparison, Mechanical, and Code Design

Complexity: ★★★

Robo's Story:

Imagine that you are going to Ikea with your family and you want to buy the perfect table for your projects at home – or maybe a box for all the toys you have. But how will we ensure that it fits in the room, between the bed and the closet?

Problem Situation:

We need to measure the distance between the objects.

Solution:

Build and program a Robo-measuring device.

Project 7: Robo Is a Measuring Device

Modules:


x2
Main Block


RGB LED


Distance Sensor


Button

Program:


Conditions Button Obstacle

Focus:

- Measuring, comparing: equal, less, greater than.
- **Robo Code App:** Button and Obstacle conditions.

Objectives:

- To build a Robo-measuring device and create a simple program that includes some of the basic Actions and the Timer and Button conditions.

Learning Outcomes:

- I know what measurement and comparison is;
- I know what a Decision Tree 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-measuring device;
- I can create a simple program that includes some of the basic Actions, Connections between them; I can modify it with the Button and Obstacle conditions.

Key vocabulary:

- Measuring, comparing, centimeter, equal, less, greater than, distance, Distance sensor;
- Robo Code App: Obstacle condition.

Project 8: Robo Is a Sound Detector

Concepts: Sound level, Sensor, Condition, Mechanical, and Code Design

Complexity: ★★☆

Robo's Story:

Imagine that you are making a surprise party for your friend and you need to know when your friend is almost home. How can you detect he is coming soon?

Problem Situation:

We need to indicate a specific sound level.

Solution:

Build and program a Robo-sound detector.

Project 8: Robo Is a Sound Detector

Modules:

Main Block

Distance Sensor

Program:

Conditions Sound

Focus:

- Decibel as a measuring unit of a sound level.
- **Robo Code App:** Sound Condition.

Objectives:

- To build a Robo-sound detector and create a simple program that includes some of the basic Actions and the Sound condition.

Learning Outcomes:

- I know and can what decibel is, how to measure sounds;
- I can consider how to combine modules to build the Robo-sound detector;
- I can create a simple program that includes some of the basic Actions, Connections between them; I can modify it with the Sound condition.

Key vocabulary:

- Sound level, decibel, sound sensor;
- Robo Code App: Sound condition.

Project 9: Robo-guard Keeps Order

Concepts: Sound level, Decision Tree, Condition, Mechanical, and Code Design

Complexity: ★★☆

Robo's Story:

Sometimes it gets too loud in a classroom and a teacher needs to ask students to be quiet. Teachers do not like these instances! How can Robo-device help the teacher in this situation?

Problem Situation:

We need to indicate a particular sound level and create a warning signal.

Solution:

Build and program a Robo-guard.

Project 9: Robo-guard Keeps Order

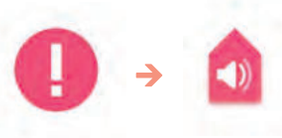
Modules:


Main Block


RGB LED


Distance Sensor

Program:


Conditions Sound

Focus:

- Comparing: equal, less, greater than, decision tree;
- **Robo Code App:** Sound Condition.

Objectives:

- To build a Robo-guard and create a simple program that includes some of the basic Actions and the Sound condition.

Learning Outcomes:

- I know and can explain what comparing is and use it to program the Sound condition;
- I know what a Decision Tree 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-guard;
- I can create a simple program that includes some of the basic Actions, Connections between them; I can modify it with the Sound condition.

Key vocabulary:

- Decibel, comparing: equal, less, greater than, Decision Tree;
- Robo Code App: Sound condition.

Project 10: Robo Is a Smart Pet

Concepts: Artificial intelligence, Decision Tree, Condition, Mechanical and Code Design **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 smart robot-pets?

Problem Situation:

You are not allowed to have a real pet, but desperately want a pet-friend to play with.

Solution:

Build and program a Robo-pet.

Project 10: Robo Is a Smart Pet

Modules:


Main Block


RGB LED


DC Motors


Servo


Distance Sensor


Button


Wheel


Connector Block

Program:


Conditions → Sound

Focus:

- Comparing: Artificial intelligence, robots-pets;
- **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 artificial intelligence is, why people create it;
- I know and can explain what a decision tree is and can use it for programming a Robo-device;
- I can consider how to combine modules to build the Robo-pet;
- I can create a simple program that includes some of the basic Actions, Connections between them; I can modify it with the Obstacle and Sound conditions.

Key vocabulary:

- Artificial intelligence, Decision Tree;
- Robo Code App: Obstacle and Sound conditions.