CODING IRELAND

# Teacher Learning Plan

## Digital Skills Curriculum 2024/25

2nd Year

# Table of Contents

# How to Use This Learning Plan

This learning plan provides an overview of all the modules available for 2nd Year, including their units, learning goals, and outcomes. Each module is designed to support both new and experienced teachers with easy-to-follow, step-by-step lessons.

## Lesson Types

There are two types of lessons in the Digital Skills Curriculum:

- **Teacher-Led Lessons** – The teacher directs and leads students through the lesson, guiding them through the activities and discussions.
- **Teacher/Student-Led Lessons** – Teachers can choose to lead the lesson, or students can follow the step-by-step instructions to work through it independently.

Younger students require a fully guided approach, while older students often benefit from working at their own pace with teacher support as needed.

## Flexible Curriculum Approach

Teachers have the flexibility to choose the modules that best fit their class needs. While there are enough lessons to cover a full school year, it is not necessary to complete all the modules. This allows teachers to tailor the learning experience to their students while ensuring they meet their educational goals.

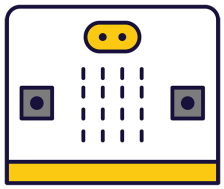
## Student Access

Students log into the platform to access their lessons. They can follow the step-by-step instructions independently, or teachers can lead the lesson as needed.

## Getting Started

1. **Review the Learning Plan:** Each module includes an overview of its goals, learning outcomes, lesson structure, and required resources. Start by familiarising yourself with the curriculum's scope.
2. **Plan Your Lessons:** Every lesson includes step-by-step guidance, accessible from your teacher dashboard. Adjust the pacing and delivery method based on your students' needs.
3. **Check Required Equipment:** Most lessons only require a laptop, Chromebook, or tablet. Some modules may include additional materials like microbits or LEDs. The required equipment is listed at the start of each module and each individual lesson.
4. **Support Student Learning:** Encourage students to work through the lessons. No prior coding experience is required—teachers can learn alongside their students.
5. **Use Assessments:** Each lesson includes a multiple-choice quiz to help assess student understanding and track progress.
6. **Need Help?:** We're always happy to answer your questions and give advice. You can contact our team at info@codingireland.ie or 01 584 9955.

# Module: Introduction to Microbit Programming



This module introduces students to the fascinating world of microbit programming. Teachers should guide students through creating new projects, exploring the project editor, and writing and deleting code. The first lesson focuses on programming microbits to display messages, react to button presses, show icons, and play melodies. The second lesson involves creating a reaction timer game, teaching students how to create variables, add random delays, and record reaction times. Teachers should encourage experimentation and exploration throughout.

| **Duration** | **Equipment** |
| --- | --- |
| 2 weeks | Students can use any of these devices:<br>• Chromebook/Laptop/PC<br><br>Required Equipment:<br>• Microbit |
| **Module Goals** | **Module Outcomes** |
| 1. Understand the fundamentals of microbits and their programming.<br>2. Develop proficiency in creating, adding, and deleting code in the project editor.<br>3. Gain skills in programming microbits to display messages, react to button presses, show icons, play melodies, and respond to movement.<br>4. Design and create a reaction timer game using a Microbit.<br>5. Learn to create variables, store time stamps, and record a player's reaction time in a game. | 1. Understand and utilise the project editor on the MakeCode for Microbit website.<br>2. Create, add, and delete code to program a Microbit to display messages, react to button presses, and play melodies.<br>3. Connect a Microbit to a computer and upload the programmed code.<br>4. Design and create a reaction timer game using a Microbit, incorporating random visual prompts.<br>5. Use variables to store time stamps and record a player's reaction time in the game. |

# Week 1

## Lesson: Exploring Microbits

| | | | |
|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz |

Prepare to introduce students to the world of microbits, a pocket-sized programmable computer. The lesson will involve creating a new project on the MakeCode for microbit website, familiarising with the project editor, and writing code to display numbers, names, and icons. Students will also learn to delete code, connect their microbits to their computers, and program their microbits to play music. The lesson concludes with an exploration phase where students can experiment with different blocks from the toolbox.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the basic functionality and features of a microbit.<br>2. Create a new project using the MakeCode for microbit website.<br>3. Use the Project Editor to write and simulate code.<br>4. Program the microbit to display numbers and text on its LED grid.<br>5. Program the microbit to respond to button presses with specific actions. | 1. Identify the functions and capabilities of a microbit.<br>2. Create a new project on the MakeCode for microbit website.<br>3. Understand the layout and functions of the Project Editor.<br>4. Write and execute code to display numbers and names on the microbit.<br>5. Program the microbit to respond to button presses with specific displays.<br>6. Connect and download code to an actual microbit device.<br>7. Compose and program a melody to play on the microbit.<br>8. Explore and experiment with different coding blocks and functions. |

# Week 2

## Lesson: Reaction Timer

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students in creating a 'Reaction Timer' project using Micro:bit. They'll start by setting up a new project, then create a welcome message and a countdown. Next, they'll add a random delay to make the game unpredictable. They'll create variables to store time stamps, and finally, record the player's reaction time. Familiarise yourself with the code snippets provided.

Students can use any of these devices (and can share if necessary):

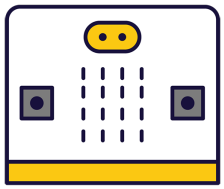
- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating and managing a new project on the Micro:bit platform.<br>2. Acquire knowledge on how to create and display messages using code.<br>3. Understand and apply the concept of countdowns and delays in programming.<br>4. Learn to create and utilise variables for storing time stamps.<br>5. Gain proficiency in recording and displaying user interactions in real-time. | 1. Develop a new project using the Micro:bit website.<br>2. Construct a welcome message to display upon powering on the Microbit.<br>3. Create a countdown sequence with visual cues using code.<br>4. Implement a random delay function in the game for unpredictability.<br>5. Create and utilise variables to store time stamps.<br>6. Record and display player reaction time upon button press. |

# Module: Applied Microbit Programming



This module delves into the world of Microbit programming, providing students with hands-on experience in creating games, sensor graphs, and IoT networks. Teachers should encourage students to experiment with different functionalities of the Microbit, such as button inputs, sensors, and radio communication. Group work is encouraged in some lessons to foster collaboration. Teachers should guide students through the coding process, ensuring understanding, and facilitate discussions on project improvements to stimulate critical and creative thinking.

| **Duration** | **Equipment** |
|---|---|
| 8 weeks | Students can use any of these devices:<br>• Chromebook/Laptop/PC<br><br>Required Equipment:<br>• Microbit |
| **Module Goals** | **Module Outcomes** |
| 1. Develop proficiency in creating interactive games using Microbit programming.<br>2. Understand and utilise various sensors on the Microbit for different applications.<br>3. Gain skills in creating multi-device IoT networks using Microbits.<br>4. Master the use of Microbit's radio feature for sending and receiving messages.<br>5. Enhance problem-solving and critical thinking skills through programming challenges. | 1. Develop a simple guessing game using Microbit, incorporating variables, button inputs, and game logic.<br>2. Investigate and understand the functionality of different sensors on the Microbit.<br>3. Create a time-based game on Microbit, enhancing precision and timing skills.<br>4. Utilise a Microbit to control a Scratch Paddle Ball game, demonstrating understanding of motion sensors.<br>5. Program a multi-device IoT network using Microbits, monitoring and displaying various environmental factors.<br>6. Enable communication between two Microbits using radio messages, demonstrating understanding of wireless communication.<br>7. Develop a motion-based game using the accelerometer on the Microbit.<br>8. Create a Microbit-based 'Invaders' game, applying advanced programming skills. |

# Week 1

## Lesson: Microbit Light Clapper

| | | | | |
|---|---|---|---|---|
| Beginner | 40 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare for the 'Microbit Light Clapper' lesson by familiarising yourself with the makecode.microbit.org website. Understand the process of creating a new project, setting up variables, and using sound thresholds. Be ready to guide students in writing code to detect claps and control LED lights. Ensure you can troubleshoot issues and explain how to test the code in the simulator and on the Microbit.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply the concept of variables in coding.<br>2. Set and utilise sound thresholds for input detection.<br>3. Implement conditional statements (if-then-else) to control LED light responses.<br>4. Test and debug code in a simulator environment.<br>5. Transfer and apply code to a physical Microbit device. | 1. Develop a new project using makecode.microbit.org.<br>2. Create and utilise a variable to control the LED lights on the Microbit.<br>3. Set a sound threshold for detecting claps using the Microbit's microphone.<br>4. Implement code to detect a clap based on the set sound threshold.<br>5. Use an 'if then else' block to control the LED lights based on the clap detection. |

# Week 2

## Lesson: Sound Recorder

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

In this lesson, students will learn how to create a new project, titled 'Music Maker'. They will be introduced to the 'music.play' function and will learn how to add music code to their project. They will also learn how to customise their melody and adjust the tempo. Finally, they will test their music by pressing the 'A' button on the simulator or on a physical Microbit device.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating a new project using a sound recorder.<br>2. Understand and apply the 'music.play' function in coding to generate melodies.<br>3. Gain knowledge in modifying tempo and designing personalised melodies.<br>4. Acquire the ability to test and troubleshoot the created music code.<br>5. Learn to transfer and utilise the coded music on a physical device. | 1. Create a new project in the sound recorder platform.<br>2. Implement the 'music.play' function to add a melody to the project.<br>3. Modify the tempo of the melody using the provided tool.<br>4. Test the created melody using the 'A' button on the simulator.<br>5. Download the code onto a physical Microbit device for offline use (optional). |

# Week 3

## Lesson: Higher or Lower Game

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare for this lesson by familiarising yourself with the Microbit project creation process and the coding involved in creating variables. Understand the game setup, including the use of random numbers and how they're displayed. Be prepared to explain the game mechanics, such as guessing higher or lower, scoring points, and resetting numbers for the next round. Be ready to guide students through the game over process and the steps to duplicate code for the 'higher' guess. Finally, ensure you know how to download the game onto a Microbit for playing.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating and manipulating variables within a Microbit project.<br>2. Understand and apply the concept of random number generation in game development.<br>3. Gain proficiency in programming button inputs to trigger specific actions.<br>4. Learn to implement scoring systems and game over conditions in a game project.<br>5. Enhance problem-solving skills by debugging and testing a game on a Microbit device. | 1. Create a new Microbit project and two variables for the game.<br>2. Set up the start of the game with random numbers and display the number on the Microbit.<br>3. Program the A button to guess if the next number is lower and score a point if the guess is correct.<br>4. Reset the variables for the next round after a correct guess and display the new number on the Microbit.<br>5. End the game if the guess is incorrect and display the final score.<br>6. Program the B button to guess if the next number is higher, following the same rules as the A button.<br>7. Download the game onto the Microbit and play it. |

# Week 4

## Lesson: Microbit Paddle Ball

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

In this lesson, students will create a Microbit Paddle Ball game using Scratch. They will learn to create a new project, add and position sprites, and make the ball bounce around the screen. They will also connect a Microbit to control the paddle, make the ball bounce off the paddle, add a backdrop, and create a game over line. The lesson concludes with programming the game over functionality and discussing potential improvements to the game. Teachers should familiarise themselves with Scratch and Microbit prior to the lesson.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating and managing a new Scratch project.<br>2. Understand and apply the concept of adding and positioning sprites in Scratch.<br>3. Gain proficiency in coding for sprite movement, interaction, and control using Scratch blocks.<br>4. Learn to integrate and use a Microbit with Scratch for real-time control of sprites.<br>5. Enhance critical thinking and problem-solving skills by identifying potential improvements to the game. | 1. Develop a new Scratch project and remove the default cat sprite.<br>2. Add and position the 'Paddle' sprite from the sprite library.<br>3. Add the 'Soccer Ball' sprite from the sprite library.<br>4. Set the X and Y coordinates to position the ball at the top center of the screen.<br>5. Code the ball to move around the screen and bounce off the edges.<br>6. Connect and configure a Microbit to the Scratch project.<br>7. Code the paddle to move left and right by tilting the Microbit.<br>8. Program the ball to bounce off the paddle when it touches it.<br>9. Add the 'Stars' backdrop from the backdrop library.<br>10. Draw a red line at the bottom of the screen for the game over line.<br>11. Code the game to end when the ball touches the red game over line.<br>12. Propose improvements to the game by adding to or changing the existing code. |

# Week 5

## Lesson: Microbit Seismic and Meteorological Station

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through programming four Microbits to create a Seismic and Meteorological Station. Each Microbit will have a unique role: monitoring temperature, light levels, and seismic activity, displaying temperature data, indicating day or night based on light levels, and alerting seismic activity. Students will use MakeCode for Microbit and learn to set up radio communication, event handlers, and display functions. Encourage students to test their projects and consider improvements such as displaying movement intensity, measuring sound levels, recording historical data, or sending alerts based on specific conditions.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the role of each Microbit in the Seismic and Meteorological Station and how they communicate with each other.<br>2. Develop skills in coding and programming Microbits to monitor and display temperature, light levels, and seismic activity.<br>3. Test and troubleshoot the coded Microbits to ensure they function as intended in the Seismic and Meteorological Station.<br>4. Apply critical thinking to improve the functionality of the Seismic and Meteorological Station.<br>5. Develop an understanding of how meteorological and seismic data can be collected, displayed, and used in real-world applications. | 1. Program four separate Microbits to perform unique roles in a Seismic and Meteorological Station.<br>2. Code the 'Seismic and Meteorological Station' Microbit to monitor and wirelessly broadcast temperature, light levels, and 'seismic' activity data.<br>3. Code the 'Temperature Display' Microbit to receive and display the temperature data from the 'Seismic and Meteorological Station' Microbit.<br>4. Code the 'Day/Night Indicator' Microbit to receive light level data and display an indication of whether it's day or night.<br>5. Code the 'Seismic Alert' Microbit to receive 'seismic' activity data and display an alert if seismic activity is detected. |

# Week 6

## Lesson: Microbit Radio Messages

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Ensure you have two Microbits and understand the concept of a radio transceiver. Familiarise yourself with the MakeCode website and how to create a new project. Understand the importance of setting a radio group for communication. Be prepared to guide students on how to send different types of messages (string, number, name and value) and how to receive and display these messages. Encourage exploration of the Radio blocks for creative coding.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and utilise the radio transceiver feature of Microbits for communication.<br>2. Create a new project using the MakeCode Microbit website.<br>3. Set and comprehend the function of radio groups in Microbit communication.<br>4. Send, receive, and display different types of messages (string, number, name-value) using Microbits.<br>5. Explore and experiment with the Radio blocks for various coding possibilities. | 1. Understand and utilise the radio transceiver feature of Microbits for communication.<br>2. Create a new project on the Microbit website and set the radio group for broadcasting.<br>3. Send and receive different types of messages (string, number, name and value) using Microbits.<br>4. Apply the code to both Microbits and test the functionality of the radio messages.<br>5. Explore and experiment with the Radio blocks for various coding possibilities. |

# Week 7

## Lesson: Microbit Voting System

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students in creating a microbit voting system. They'll create two projects on the MakeCode Microbit website, one for voting microbits and another for a central microbit. They'll program the A and B buttons to cast votes, set up the central microbit to receive votes and reset the system, and display the vote results. They'll also enhance the system with a security feature. Ensure students understand the coding involved and the importance of testing their system.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop a microbit voting system with separate voting and central microbits.<br>2. Programme the voting microbits to cast a single 'Yes' or 'No' vote.<br>3. Configure the central microbit to receive votes, count them, and reset the voting system.<br>4. Implement a reset function on the voting microbits to allow for multiple rounds of voting.<br>5. Enhance the voting system by adding a security feature to ensure the integrity of the votes. | 1. Develop two separate projects on the MakeCode Microbit website for voting and central microbits.<br>2. Programme the A and B buttons on the microbit to cast a single 'Yes' or 'No' vote.<br>3. Set up the central microbit to receive votes, count them, and reset the voting system when needed.<br>4. Configure the individual voting microbits to receive the 'Reset' signal from the central microbit.<br>5. Display the vote results on the central microbit. |

# Week 8

## Lesson: Microbit Lab

| | | |
|---|---|---|
| Advanced | 60 mins | Teacher led |

Prepare to introduce the Microbit Lab lesson, demonstrating a simple Microbit project to inspire students. Divide students into groups for brainstorming and project creation. Facilitate brainstorming, feedback, and project creation sessions, ensuring students keep their ideas simple and achievable. Encourage constructive feedback and teamwork. Finally, organise a 'Show and Tell' session for groups to present their projects, fostering a supportive learning environment and reinforcing the importance of teamwork.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop a simple Microbit project using basic blocks.<br>2. Work effectively in groups, contributing ideas and making collective decisions.<br>3. Present and explain a project idea, including its components and envisioned outcome.<br>4. Give and receive constructive feedback, and incorporate it into project plans.<br>5. Code a Microbit project, demonstrating problem-solving and creativity. | 1. Brainstorm and develop a simple Microbit project idea in a group setting.<br>2. Present the project idea to the class, explaining the planned LED patterns and inputs.<br>3. Incorporate feedback from peers and teacher into the project plan.<br>4. Create a Microbit project based on the brainstormed idea and feedback received.<br>5. Present the final Microbit project to the class, explaining the coding process, changes made, and learnings from the process. |

# Module: Introduction to HTML and CSS

This module introduces students to HTML and CSS, the fundamental languages for creating and styling web pages. Teachers should guide students through the structure of a web page, basic HTML elements, tables, lists, form creation, and multimedia embedding. The module then transitions to CSS, covering topics such as CSS rules, selectors, the box model, text and font styling, and website layout. Practical exercises are included throughout to reinforce learning and encourage hands-on practice. Teachers should ensure students understand the concepts and encourage them to experiment with different code values.

| **Duration** | **Equipment** |
| --- | --- |
| 9 weeks | Students can use any of these devices:<br>• Chromebook/Laptop/PC<br>• iPad/Tablet |
| **Module Goals** | **Module Outcomes** |
| 1. Understand and apply basic and advanced HTML elements to create structured web pages.<br>2. Create and manipulate tables and lists in HTML for effective data presentation.<br>3. Design interactive forms using basic and advanced HTML input types.<br>4. Embed multimedia elements into web pages using HTML5.<br>5. Apply CSS for styling web pages, including text, fonts, and layout design. | 1. Understand and apply basic HTML elements such as headings, paragraphs, breaks, images, and links to structure a webpage.<br>2. Create simple and complex tables in HTML, utilising advanced features like rowspan and colspan.<br>3. Design and code interactive forms using HTML elements, including advanced input types.<br>4. Embed audio and video files into web pages using HTML5 multimedia elements.<br>5. Apply CSS to style web pages, including understanding the CSS Box Model, styling text, manipulating fonts, and creating website layouts. |

# Week 1

## Lesson: Introduction to HTML

| | | | |
|---|---|---|---|
| Beginner | 35 mins | Teacher/Student led | Student Quiz |

Prepare to introduce HTML as the standard markup language for creating web pages. Explain the structure of a web page. Discuss HTML elements, their start tags, content, and end tags. Highlight how web browsers interpret HTML code to display web pages. Facilitate hands-on practice with writing basic HTML code and adding content. Encourage students to experiment with their own details.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the purpose and structure of HTML in web development.<br>2. Identify and describe the key components of a web page structure.<br>3. Recognise and use basic HTML elements in coding.<br>4. Explain the role of web browsers in interpreting and displaying HTML code.<br>5. Apply knowledge to write and modify basic HTML code to create a simple web page. | 1. Define HTML and its role in web page creation.<br>2. Identify and explain the structure of a web page using HTML.<br>3. Recognise and describe HTML elements and their functions.<br>4. Understand how web browsers interpret and display HTML code.<br>5. Write and run basic HTML code to create a simple web page. |

## Lesson: HTML Basic Elements

| | | | |
|---|---|---|---|
| Beginner | 30 mins | Teacher/Student led | Student Quiz |

Prepare to guide students through understanding basic HTML elements. Start with explaining heading tags, their importance and usage. Move on to defining paragraphs, line breaks, and how to incorporate images with attributes. Discuss the concept of links, their attributes and how to create them. Finally, encourage students to write their own HTML code using these elements. Ensure to provide real-time examples and encourage hands-on practice.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply HTML heading tags from h1 to h6.<br>2. Create and format paragraphs using the p tag.<br>3. Implement line breaks in HTML text using the br tag.<br>4. Insert and manipulate images using the img tag and its attributes.<br>5. Create hyperlinks to other web pages using the a tag and its attributes. | 1. Identify and use HTML heading tags from <h1> to <h6>.<br>2. Define and implement paragraphs using the <p> tag.<br>3. Insert line breaks in HTML documents using the <br> tag.<br>4. Embed images in HTML using the <img> tag and its attributes.<br>5. Create hyperlinks using the <a> tag and understand the use of its attributes. |

# Week 2

## Lesson: HTML Tables

| | | | |
|---|---|---|---|
| Beginner | 30 mins | Teacher/Student led | Student Quiz |

Prepare to guide students through the process of creating HTML tables. Begin with an introduction to tables and their structure, including headers, bodies, and footers. Then, delve into the specific HTML tags used to create tables, rows, and cells. Provide examples and encourage students to practice coding their own tables. Finally, challenge students to add multiple rows to their tables. Ensure students understand the importance of correctly nesting tags within each other.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the purpose and structure of HTML tables.<br>2. Identify and use HTML tags to create table headers, bodies, and footers.<br>3. Create table rows and columns using appropriate HTML tags.<br>4. Apply HTML coding to create a personalised table with multiple rows and columns.<br>5. Develop skills to troubleshoot and correct HTML table coding errors. | 1. Identify and explain the structure and purpose of HTML tables.<br>2. Create a basic HTML table using the <table>, <tr>, and <td> tags.<br>3. Use <thead>, <tbody>, and <tfoot> tags to define the header, body, and footer of a table.<br>4. Develop a multi-row HTML table with specific content in each cell.<br>5. Apply CSS styling to HTML tables and their cells. |

## Lesson: Crafting Complex Tables

| | | | |
|---|---|---|---|
| Intermediate | 30 mins | Teacher/Student led | Student Quiz |

Prepare to guide students through the process of crafting complex HTML tables. Ensure they understand basic table creation before introducing headers and footers. Highlight the importance of the 'rowspan' and 'colspan' attributes for merging cells. Encourage experimentation with these attributes to see their effects. Finally, review the completed table to ensure understanding.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Master the creation of basic HTML tables with multiple rows and columns.<br>2. Understand and apply the concept of table headers for better data context.<br>3. Learn to use the 'rowspan' attribute to merge cells vertically.<br>4. Learn to use the 'colspan' attribute to merge cells horizontally.<br>5. Develop the ability to add footer rows to tables for summary information. | 1. Construct a basic HTML table with multiple rows and columns.<br>2. Integrate a header row into the table for column labelling.<br>3. Apply the 'rowspan' attribute to merge cells vertically.<br>4. Utilise the 'colspan' attribute to merge cells horizontally.<br>5. Add a footer row to the table for summary information. |

# Week 3

## Lesson: HTML Lists

| | | | |
|---|---|---|---|
| Intermediate | 30 mins | Teacher/Student led | Student Quiz |

Prepare to introduce HTML lists, distinguishing between ordered and unordered types. Explain the use of <ol></ol> and <li></li> tags for ordered lists, and the <ul></ul> and <li></li> tags for unordered lists. Discuss the 'type' attribute for customising list markers. Demonstrate nested lists and encourage students to code their own lists, experimenting with different types and nesting.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the purpose and usage of HTML lists.<br>2. Code ordered and unordered lists in HTML.<br>3. Manipulate the numbering of ordered lists and bullet points of unordered lists.<br>4. Create nested lists in HTML.<br>5. Apply learned skills to code a personal list. | 1. Identify and differentiate between ordered and unordered HTML lists.<br>2. Code ordered lists using the <ol> and <li> tags.<br>3. Manipulate the numbering type of ordered lists using the 'type' attribute.<br>4. Code unordered lists using the <ul> and <li> tags.<br>5. Alter the bullet point type of unordered lists using the 'type' attribute.<br>6. Create nested lists within both ordered and unordered lists.<br>7. Apply learned skills to code a personalised list. |

# Week 4

## Lesson: Basics of Form Creation

| | | | |
|---|---|---|---|
| 🟡 Intermediate | 🕒 60 mins | 👥 Teacher/Student led | ☑ Student Quiz |

In this lesson, teachers will guide students through the process of creating a basic HTML form. Starting with an introduction to HTML forms, students will learn how to create a form container, add input fields, labels, a textarea, and a submit button. The lesson encourages experimentation and exploration, allowing students to modify the code and observe the changes. Teachers should emphasise the importance of each element and attribute in the form, and how they contribute to the overall functionality and accessibility of the form.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the purpose and structure of HTML forms<br>2. Create a form container using the <form> element<br>3. Add and configure input fields for user data collection<br>4. Implement labels for enhanced accessibility and usability<br>5. Include a textarea for multi-line text input<br>6. Add a submit button to enable form submission | 1. Understand and apply the HTML <form> element to create a form container.<br>2. Create and utilise <input> fields for text and email data collection.<br>3. Implement <label> elements for improved form accessibility and usability.<br>4. Add a <textarea> element for multi-line text input.<br>5. Include a <button> with a type attribute set to submit to finalise form creation. |

## Lesson: Advanced Input Types

| | | | |
|---|---|---|---|
| 🔴 Advanced | 🕒 60 mins | 👥 Teacher/Student led | ☑ Student Quiz |

This lesson explores advanced HTML form input types: number, date, and colour. Teachers should guide students through the creation of number, date, and colour input fields, demonstrating the enhanced functionality and user experience these types offer. The lesson concludes with students enhancing a form with these advanced input types, applying their newly acquired knowledge.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
| --- | --- |
| 1. Understand and utilise advanced HTML input types including number, date, and colour.<br>2. Create a number input field for capturing numerical data such as age.<br>3. Implement a date input field for selecting specific dates.<br>4. Use a colour input field for selecting a colour from a colour picker.<br>5. Enhance a form by integrating advanced input types to improve functionality and user experience. | 1. Understand and apply the number input type in HTML forms.<br>2. Understand and apply the date input type in HTML forms.<br>3. Understand and apply the color input type in HTML forms.<br>4. Create HTML forms utilising advanced input types.<br>5. Enhance user experience by implementing advanced input types in forms. |

# Week 5

## Lesson: Embedding Audio and Video

| | | | |
|---|---|---|---|
| Advanced | 30 mins | Teacher/Student led | Student Quiz |

In this lesson, teachers will guide students through the process of embedding audio and video into a webpage using HTML5. They will start by discussing the importance of understanding supported formats for different web browsers. Students will then learn how to embed audio and video files using the HTML5 <audio> and <video> tags. Teachers will encourage students to test their work and reflect on their learning. The lesson will conclude with a review and encouragement for continued practice.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Identify and understand the different audio and video formats supported by HTML5.<br>2. Embed an audio file into a web page using the HTML5 <audio> tag.<br>3. Embed a video file into a web page using the HTML5 <video> tag.<br>4. Preview and test the functionality of embedded audio and video players.<br>5. Reflect on the process and importance of embedding multimedia elements in web pages. | 1. Identify and understand the common audio and video formats supported by HTML5 and their compatibility with various web browsers.<br>2. Embed an audio file into a web page using the HTML5 <audio> tag and provide an MP3 file as the source.<br>3. Embed a video file into a web page using the HTML5 <video> tag and provide an MP4 file as the source.<br>4. Preview and test the functionality of the embedded audio and video players, and experiment with different video dimensions.<br>5. Reflect on the process of embedding audio and video files into a web page using HTML5 tags and understand the importance of compatibility and user experience. |

# Week 6

## Lesson: Introduction to CSS

| | | | |
|---|---|---|---|
| Advanced | 30 mins | Teacher/Student led | Student Quiz |

Prepare to introduce students to CSS, the language for styling web pages. Explain what CSS is and how it interacts with HTML. Discuss CSS rules, selectors, and declaration blocks. Use examples to illustrate how CSS changes the appearance of HTML elements. Introduce the concept of element, ID, and class selectors. Explain the style property and how it can be used to directly apply CSS to HTML elements. Provide exercises for students to practice writing CSS code and applying different selectors.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the purpose and function of CSS in web development.<br>2. Identify and apply CSS rules including selectors and declaration blocks.<br>3. Differentiate between element, ID, and class selectors in CSS.<br>4. Apply CSS styles directly to HTML elements using the style property.<br>5. Practice creating and applying CSS classes to HTML elements. | 1. Understand the purpose and function of CSS in web development.<br>2. Identify and apply CSS rules including selectors and declaration blocks.<br>3. Use CSS to style HTML elements using element, ID, and class selectors.<br>4. Apply CSS properties directly to HTML elements using the style property.<br>5. Practice writing CSS code through exercises and understand how it affects the appearance of HTML elements. |

## Lesson: CSS Box Model

| | | | |
|---|---|---|---|
| Advanced | 30 mins | Teacher/Student led | Student Quiz |

Prepare to guide students through understanding the CSS Box Model, including margins, borders, padding, and content. Demonstrate how to apply different border styles, widths, and colours. Encourage students to experiment with padding and margins to understand their impact on layout. The lesson includes practical exercises to reinforce learning. Ensure students understand how to use the code examples provided.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the CSS Box Model and its components: margins, borders, padding, and content.<br>2. Apply different styles, widths, and colours to CSS borders.<br>3. Manipulate padding in CSS to create space around content within the border.<br>4. Use CSS margins to create space around elements, outside the border.<br>5. Perform exercises to apply CSS Box Model properties to HTML elements. | 1. Identify and explain the components of the CSS Box Model.<br>2. Apply CSS properties to create and modify borders on HTML elements.<br>3. Use CSS properties to set border styles, widths, and colours.<br>4. Apply CSS padding to create space around content within an element.<br>5. Use CSS margins to create space around HTML elements. |

# Week 7

## Lesson: CSS Text

| | | | |
|---|---|---|---|
| Advanced | 40 mins | Teacher/Student led | Student Quiz |

Prepare to guide students through the process of styling text using CSS. The lesson covers setting text and background colours, alignment, decoration, transformation, spacing, and adding a shadow. Students will learn to use properties such as 'color', 'text-align', 'text-decoration', 'text-transform', 'letter-spacing', 'word-spacing', 'line-height', 'text-indent', and 'text-shadow'. They will also experiment with different values for these properties, including colour names, HEX values, RGB values, and pixel sizes.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply CSS properties to style text colour and background colour.<br>2. Manipulate text alignment using CSS.<br>3. Use CSS to add or remove text decorations.<br>4. Transform text to uppercase, lowercase or capitalised format using CSS.<br>5. Apply CSS properties to adjust text spacing and add text shadow. | 1. Apply CSS properties to style text colour and background colour.<br>2. Align text using CSS properties.<br>3. Decorate text using underline, overline and line-through CSS properties.<br>4. Transform text to uppercase, lowercase and capitalise using CSS properties.<br>5. Apply CSS properties to set text spacing and add text shadow. |

## Lesson: CSS Fonts

| | | | |
|---|---|---|---|
| Advanced | 40 mins | Teacher/Student led | Student Quiz |

Prepare to introduce students to CSS fonts, explaining their importance in web design. Discuss the 'font-family' property and provide examples of commonly used font families. Explain the concept of 'web safe fonts' and the use of fallback fonts. Introduce 'font-weight', 'font-size', and 'font-style' properties. Prepare an exercise where students will code a paragraph of text using the discussed properties. Be ready to provide a solution and explain the code.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the importance of font selection in CSS and how it impacts user experience.<br>2. Apply the 'font-family' property to set specific fonts for text elements.<br>3. Use 'font-family' to specify fallback fonts when the primary font is unavailable.<br>4. Manipulate 'font-weight', 'font-size', and 'font-style' properties to modify the appearance of text.<br>5. Implement learned CSS font properties in a practical exercise. | 1. Apply the CSS property 'font-family' to set specific fonts for text.<br>2. Specify 'fallback' fonts using the 'font-family' CSS property.<br>3. Manipulate the weight of the font using the 'font-weight' CSS property.<br>4. Adjust the size of the font using the 'font-size' CSS property.<br>5. Change the style of the font using the 'font-style' CSS property. |

# Week 8

## Lesson: CSS Website Layout

| | | | |
|---|---|---|---|
| Advanced | 40 mins | Teacher/Student led | Student Quiz |

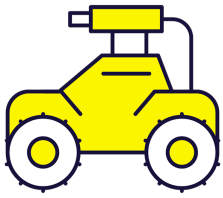
This lesson focuses on CSS website layout. Teachers should familiarise themselves with the basic structure of a website, including the header, content, and footer. The lesson covers how to code these areas using HTML and CSS, with practical examples provided. It also explores different content layouts, such as one-column, two-column, and three-column layouts. The lesson concludes with a comprehensive example of putting all the elements together to create a complete website layout. Teachers should encourage students to experiment with the code examples provided.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the structure of a website layout including header, content, and footer.<br>2. Develop skills to code and style a website header using HTML and CSS.<br>3. Learn to create different content layouts such as one-column, two-column, and three-column layouts.<br>4. Gain proficiency in using CSS to set column widths and layout.<br>5. Acquire knowledge to code and style a website footer using HTML and CSS. | 1. Identify and code common website layout areas: header, content, and footer using CSS.<br>2. Construct and style a website header with logo and navigation menu.<br>3. Design and implement one, two, and three column content layouts.<br>4. Manipulate column widths and padding to achieve desired layout.<br>5. Create and style a website footer with company information and secondary links.<br>6. Combine all elements to create a cohesive website layout. |

# Module: Driving Innovation with Microbits and Cars



This module enables students to explore the innovative use of Microbits and cars. It begins with building and programming traffic lights, followed by constructing a Move Motor Sensor Car. Students then learn to program the car to follow lines, measure distances, and navigate around objects. The module culminates in the assembly and programming of a Move Motor Klaw. Teachers should encourage experimentation and creativity throughout, and provide guidance during complex assembly processes.

| **Duration** | **Equipment** |
|---|---|
| 9 weeks | Students can use any of these devices:<br>• Chromebook/Laptop/PC<br><br>Required Equipment:<br>• Microbit<br>• Move Motor Car<br>• Move Motor Klaw<br>• Phillips Screwdriver<br>• Traffic Lights Kit |
| **Module Goals** | **Module Outcomes** |
| 1. Master the assembly and programming of Microbit Traffic Lights Kit.<br>2. Understand and implement the construction and programming of Move Motor Sensor Car.<br>3. Develop skills to program the car to follow lines and navigate around objects using sensors.<br>4. Learn to use the accelerometer and radio in Microbit for remote control of the car.<br>5. Gain proficiency in assembling and programming the Move Motor Klaw for various applications. | 1. Construct and program Microbit traffic lights to execute a sequence.<br>2. Assemble and program a Move Motor Sensor Car with Microbit.<br>3. Program a line-following car using sensor values and conditional statements.<br>4. Utilise the sonar sensor on the Move Motor Car to measure distances and transmit the results to another Microbit.<br>5. Program distance sensors on the Move Motor Car to detect and navigate around objects. |

# Week 1

## Lesson: Build your Traffic Lights

| | | | |
|---|---|---|---|
| Beginner | 10 mins | Teacher/Student led | Student Quiz |

Ensure students have all necessary materials, including the Microbit Traffic Lights Kit, a Microbit, and a Phillips head screwdriver. Guide them through opening the package and assembling the stand. Assist them in correctly positioning the Microbit on the traffic lights, ensuring they align the holes correctly. Supervise as they use the screwdriver to secure the Microbit. Celebrate their accomplishment once completed.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit
- Phillips Screwdriver

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Identify and gather necessary components for the Microbit Traffic Lights Kit.<br>2. Understand and execute the process of unpacking and preparing the kit.<br>3. Develop skills in assembling the stand for the traffic lights.<br>4. Apply knowledge of Microbit to correctly align and attach it to the traffic lights.<br>5. Demonstrate the ability to follow step-by-step instructions to complete a technical task. | 1. Identify and gather necessary components for the Microbit Traffic Lights Kit.<br>2. Unpack and organise the Microbit Traffic Lights package contents.<br>3. Assemble the stand from the provided parts in the kit.<br>4. Align and attach the Microbit to the traffic lights using the correct hole configuration.<br>5. Successfully complete the assembly of the Microbit Traffic Lights. |

## Lesson: Microbit Traffic Lights

| | | | | |
|---|---|---|---|---|
| Beginner | 40 mins | Teacher/Student led | Student Quiz | Student Challenge |

In this lesson, students will create a new Microbit project on makecode.com, add the Stopbit extension, and test all the lights. They will learn about sequences in coding and apply this knowledge to program a traffic light sequence using on/off and state methods. Students will need to check the correct display of lights in each sequence.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit

| Learning Goals | Learning Outcomes |
| --- | --- |
| 1. Understand and apply the process of creating a new Microbit project.<br>2. Learn to add and utilise the Stopbit extension for programming traffic lights kit.<br>3. Gain skills in testing and troubleshooting the functionality of the lights.<br>4. Comprehend the concept of 'sequence' in coding and apply it to program traffic lights.<br>5. Develop proficiency in programming the sequence of traffic lights using on/off and state methods. | 1. Create and manage a new Microbit project on makecode.com.<br>2. Add and utilise the "stopbit" extension to the Microbit project.<br>3. Test and troubleshoot the functionality of each light on the Microbit.<br>4. Understand and apply the concept of 'sequence' in coding to program traffic lights.<br>5. Program the sequence of traffic lights using on/off and state methods. |

# Week 2

## Lesson: Build your Move Motor Sensor Car

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Ensure all materials are ready, including the Microbit and 4 AA batteries. Guide students through the step-by-step instructions provided in the yellow booklet, ensuring they understand each stage of assembly, connection to Makecode, and adding the Move Motor Extension. Facilitate their understanding of coding the motors, using the buzzer, Zip LEDs, line following sensors, and the distance sensor. Encourage exploration and experimentation once the Move Motor Sensor Car is built.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop practical skills in assembling a Move Motor Sensor Car.<br>2. Understand how to connect the Move Motor Sensor Car to Makecode.<br>3. Acquire coding skills for controlling the motors, buzzer, and LEDs of the Move Motor Sensor Car.<br>4. Learn to utilise the line following and distance sensors for navigation.<br>5. Encourage exploration and creativity in coding for different movements and LED usage. | 1. Identify and organise components of the Move Motor Sensor Car kit.<br>2. Assemble the Move Motor Sensor Car following the provided instructions.<br>3. Connect the assembled car to Makecode and add the Move Motor Extension.<br>4. Code the motors, buzzer, Zip LEDs, line following sensors, and distance sensor of the car.<br>5. Apply learned skills to explore and create new movements and LED patterns. |

# Week 3

## Lesson: Line Following Car

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

In this lesson, students will program a Move Motor Car to follow a line track using a Microbit. They will create a new project on the MakeCode website, add the kitronik-move-motor extension, and create variables for the left and right line sensors and their difference. Students will then program the car to turn right, left, and move forward based on these sensor readings. After testing their code on a track, students can tweak the code to improve the car's speed and performance on more complex tracks.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply the concept of programming a Microbit to control a Move Motor Car.<br>2. Create and manipulate variables to store sensor values and control the car's movements.<br>3. Implement conditional logic to guide the car's movements based on sensor readings.<br>4. Use LEDs for visual feedback and enhance the functionality of the car.<br>5. Experiment with code modifications to optimise the car's performance on different tracks. | 1. Programme the Move Motor Car to follow a line track using a Microbit.<br>2. Create a new project on the https://makecode.microbit.org website.<br>3. Add the kitronik-move-motor extension to the project and utilise the custom blocks to program the Move Motor car.<br>4. Create and utilise variables to store values of the left and right line sensors and their difference.<br>5. Set up the LEDs on the Move Motor car to light up different colours depending on the car's direction.<br>6. Programme the car to turn right when the left sensor reads a higher darker value than the right sensor.<br>7. Programme the car to turn left when the right sensor reads a higher darker value than the left sensor.<br>8. Programme the car to move forwards when the left and right sensors have similar readings.<br>9. Test the programmed car on a track and observe its autonomous driving.<br>10. Tweak the code to improve the car's speed and performance on different tracks. |

# Week 4

## Lesson: Move Motor Measure

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

This lesson involves using a sonar sensor on a Move Motor car to measure distances and transmit the data to another Microbit. Students will learn how an ultrasonic sensor works and how to program two Microbits to communicate with each other. They will need to add specific extensions to their project, set units and radio groups, and create code to measure and display distances. The lesson requires hands-on work with Microbits and the Move Motor car, and also includes online coding using the makecode.microbit.org website.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the function and operation of an ultrasonic sensor.<br>2. Develop proficiency in programming Microbits for specific tasks.<br>3. Gain skills in using radio groups for communication between Microbits.<br>4. Learn to measure distances using an ultrasonic sensor and a Microbit.<br>5. Acquire the ability to display measurements on a separate Microbit. | 1. Understand and explain the function of an ultrasonic sensor and how it measures distance.<br>2. Program the Microbit inside the Move Motor car to measure distances and send the measurements to another Microbit.<br>3. Add necessary extensions to the project and set the units for measurement.<br>4. Program the second Microbit to send a "measure" message and display the received measurement.<br>5. Successfully test the functionality of the system by measuring and displaying distances. |

# Week 5

## Lesson: Car Distance Sensors

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to introduce students to the concept of ultrasonic sensors and how they function. Guide them through creating a new project on the MakeCode website, adding the kitronik-move-motor extension. Assist them in programming the sensor to measure distance and display it on the Microbit. Progress to programming the car to maintain a 10cm distance from an object, including reversing. Finally, challenge students to improve the code, adding lights and randomised movement to enhance the car's obstacle avoidance.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the function and operation of ultrasonic sensors.<br>2. Develop skills in creating a new project using the kitronik-move-motor extension.<br>3. Acquire the ability to program a sensor to measure distance.<br>4. Learn to code a car to maintain a specific distance from an object.<br>5. Enhance problem-solving skills by programming the car to reverse and maintain distance. | 1. Understand the function and operation of an ultrasonic sensor.<br>2. Create a new project on the MakeCode Microbit website and add the necessary extension.<br>3. Program the sensor to measure and display the distance to an object.<br>4. Modify the code to make the car maintain a distance of 10cm from an object.<br>5. Enhance the code to reverse the car until it is exactly 10cm away from an object.<br>6. Program the car to free roam and avoid objects by stopping, reversing, and turning right when an object is detected within 10cm.<br>7. Improve the code for better navigation and add lights for visual feedback. |

# Week 6

## Lesson: Tilt Remote Control Car

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

In this lesson, students will learn to control a Move Motor car using a Microbit as a remote controller. They will create two code projects: one for the remote control and another for the car. The lesson involves programming the Microbit to detect tilts in different directions and send corresponding messages to the car. The students will also add code to stop the car and to light up the LEDs on the car in different colours. Ensure each remote and car set uses a different radio group to avoid crossed signals.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply the concept of radio communication between two Microbits.<br>2. Programme a Microbit to send specific messages based on different gestures.<br>3. Develop the ability to programme a Move Motor car to respond to different messages received.<br>4. Test and debug the code to ensure the car responds correctly to the remote control.<br>5. Extend the project by adding additional features such as LED light changes. | 1. Programme a Microbit as a remote control to send directional commands.<br>2. Programme a Microbit to receive and execute directional commands in a Move Motor car.<br>3. Test and debug the code to ensure correct functioning of the remote-controlled car.<br>4. Download and implement the code onto the Microbits.<br>5. Extend the code to include LED light changes in response to different commands as an additional challenge. |

# Week 7

## Lesson: Traffic Lights and Car Communication

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

This lesson involves coding a set of traffic lights and a robot car using Microbits. Students will program the traffic lights to display a sequence and broadcast the light being shown. The robot car will receive this broadcast and decide whether to stop or go. The lesson involves creating two code projects, adding a 'stopbit' extension, programming a sequence, broadcasting the state, programming the car, receiving the message, downloading the code, and an additional challenge. Teachers should ensure they have the necessary equipment and familiarise themselves with the coding platforms used.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Traffic Lights Kit
- Move Motor Car

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply the concept of radio communication between Microbits.<br>2. Program a sequence of traffic light signals using code blocks.<br>3. Develop skills to broadcast and receive specific messages based on traffic light states.<br>4. Control the movement of a robot car based on received messages.<br>5. Enhance problem-solving skills by modifying the code to respond based on the proximity of the car to the traffic lights. | 1. Code a set of traffic lights to run through a sequence and broadcast the displayed light.<br>2. Program a robot car to receive the broadcast and decide whether to stop or go based on the traffic light signal.<br>3. Use the "stopbit" extension to create custom code blocks for programming the traffic lights kit.<br>4. Program the car to move at different speeds or stop, depending on the received message from the traffic lights.<br>5. Modify the code to make the car respond to the traffic lights based on its proximity to them. |

# Week 8

## Lesson: Attach the Move Motor Klaw

| | | | |
|---|---|---|---|
| Advanced | 30 mins | Teacher/Student led | Student Quiz |

Ensure to guide students in seeking adult assistance for the complex parts of constructing the Move Motor Sensor Car. Facilitate the unpacking process, ensuring all necessary items are present. Lastly, guide students through the instruction booklet, assisting them in attaching the Move Motor Klaw to the car, either vertically or horizontally.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car
- Phillips Screwdriver
- Move Motor Klaw

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop the ability to follow complex instructions with adult supervision.<br>2. Understand the process of unpacking and organising components for assembly.<br>3. Gain practical skills in using tools such as a small Phillips head screwdriver.<br>4. Learn to assemble the Move Motor Klaw and attach it to the Move Motor Car.<br>5. Understand the flexibility of design in attaching the Klaw either vertically or horizontally. | 1. Identify and gather necessary components for building the Move Motor Klaw.<br>2. Correctly open the package and organise its contents.<br>3. Follow the provided instructions to assemble the Move Motor Klaw.<br>4. Successfully attach the Move Motor Klaw to the Move Motor Car.<br>5. Demonstrate safe and effective use of tools during assembly. |

## Lesson: Robot Car Claw

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Ensure you have a Move Motor Car with a Move Motor Klaw and a Microbit before starting. Create a new project on the Microbit website and add the kitronik-move-motor extension. Understand how servos work and how they control the claw's pinchers. Program the claw to close and open using specific codes and test these functions. Create a variable to vary the amount the claw closes and re-program the buttons to gradually open and close the claw. Encourage students to explore other programming possibilities with the claw.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- Move Motor Car
- Move Motor Klaw

| Learning Goals | Learning Outcomes |
| --- | --- |
| 1. Understand the setup and requirements for a Move Motor Car with a Move Motor Klaw and a Microbit.<br>2. Develop skills in creating a new Microbit project using the provided website.<br>3. Gain knowledge about the kitronik-move-motor extension and how to add it to the project.<br>4. Comprehend the functioning of servos and their application in controlling the Move Motor Klaw.<br>5. Acquire the ability to program the claw to open and close using specific code blocks. | 1. Assemble a Move Motor Car with a Move Motor Klaw and a Microbit.<br>2. Create a new Microbit project on the specified website.<br>3. Add the kitronik-move-motor extension to the project.<br>4. Understand the function and operation of a servo in the Move Motor Klaw.<br>5. Program the claw to close and open using specific code blocks and test its functionality.<br>6. Modify the code to allow the claw to open and close gradually.<br>7. Explore other potential programming and usage possibilities for the claw. |

# Module: Game Design Essentials

This module guides students through creating various interactive games using MakeCode Arcade. Each week focuses on a different project, teaching students to design sprites, control movements, program interactions, and set up game mechanics. Teachers should ensure students understand each step before moving on, encourage experimentation with the code, and emphasise the importance of correct variable selection and code placement. The module concludes with a game showcase, allowing students to present their creations.

| **Duration** | **Equipment** |
| --- | --- |
| 8 weeks | Students can use any of these devices:<br>• Chromebook/Laptop/PC<br>• iPad/Tablet |
| **Module Goals** | **Module Outcomes** |
| 1. Master the use of MakeCode Arcade for game design and development.<br>2. Develop skills in creating and controlling game sprites, including player and AI-controlled characters.<br>3. Understand and implement game mechanics such as scoring, lives, collision effects, and game outcomes.<br>4. Apply coding concepts to create interactive games with different themes and mechanics.<br>5. Gain the ability to design, code, test, and refine a variety of games, culminating in a final game showcase. | 1. Create and control game sprites using MakeCode Arcade.<br>2. Design and implement game mechanics such as movement, collision detection, scoring, and game over conditions.<br>3. Understand and apply coding concepts to create interactive games, including sprite overlaps, game logic, and variable tracking.<br>4. Develop a variety of games including arcade, platform, and battle arena games, demonstrating creativity and technical skills.<br>5. Present a completed game project, demonstrating understanding of game design principles and coding concepts. |

# Week 1

## Lesson: First Arcade Project

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

This lesson guides students through creating their first arcade project using MakeCode Arcade. They will learn about the code editor, how to create a new project, add a sprite, choose a sprite from the gallery, move the sprite, draw a tile map, draw walls, make the camera follow the sprite, add projectiles, set their direction and speed, detect overlap, lose a life, and finally, send the code to a handheld device. The lesson is hands-on and interactive, allowing students to learn by doing.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and utilise MakeCode Arcade for creating games.<br>2. Manipulate the Code Editor to build and modify game elements.<br>3. Create and customise sprites for use in a game.<br>4. Develop a tile map and implement walls for game navigation.<br>5. Implement game mechanics such as projectiles, sprite movement, and life count. | 1. Understand the functions and features of MakeCode Arcade.<br>2. Use the MakeCode Arcade code editor to create a new project and add a sprite.<br>3. Manipulate the sprite's movements using the direction buttons in the simulator.<br>4. Create and edit a tile map, including drawing walls and setting the camera to follow the sprite.<br>5. Design and implement projectiles, including setting their direction and speed, and programming responses to overlaps with the player's sprite. |

# Week 2

## Lesson: Space Dodge

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through creating a 'Space Dodge' game using MakeCode Arcade. Students will learn to create a new project, design a spaceship sprite, control the spaceship, set the number of lives, create asteroids, set their position and velocity, auto destroy them when they move off the screen, and detect when an asteroid hits the spaceship. Ensure students understand the importance of correct variable selection and the effect of different values in the code.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in using MakeCode Arcade to create a game.<br>2. Understand and apply the concept of sprites in game development.<br>3. Implement controls for game characters using code.<br>4. Apply the concept of randomisation in game elements for varied gameplay.<br>5. Understand and implement game mechanics such as collision detection and life count. | 1. Design and create a spaceship sprite using MakeCode Arcade.<br>2. Control the spaceship's movement with arrow keys and set boundaries to prevent it from going off the screen.<br>3. Set the number of lives for the spaceship.<br>4. Create and design asteroid sprites that appear at random positions on the screen.<br>5. Set the velocity of the asteroids to make them move across the screen.<br>6. Implement a function to auto-destroy asteroids when they move off the screen.<br>7. Program the game to detect when an asteroid hits the spaceship, causing it to lose a life and trigger a camera shake effect. |

# Week 3

## Lesson: Bat Battle

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

This lesson guides students through creating a game using MakeCode Arcade. They will learn to create and control a player sprite, generate enemy sprites, and program interactions between them. The lesson includes coding for scoring points and ending the game. Teachers should ensure students understand each step before moving on, and encourage experimentation with the code to add new features to the game.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in using MakeCode Arcade to create an interactive game.<br>2. Understand how to create, control, and position player and enemy sprites.<br>3. Learn to program game interactions such as shooting projectiles and detecting overlaps.<br>4. Gain knowledge on how to keep score and end the game in MakeCode Arcade.<br>5. Enhance problem-solving and debugging skills by experimenting with the code and adding new features. | 1. Create and control a player sprite in MakeCode Arcade.<br>2. Generate enemy sprites at random positions.<br>3. Program interactions between player and enemy sprites.<br>4. Implement a scoring system for hitting targets.<br>5. End the game when an enemy sprite hits the player sprite. |

# Week 4

## Lesson: Space Shooter

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through creating a space-themed game using MakeCode Arcade. They will design a spaceship sprite, control its movements, set the number of lives, create and program asteroids, fire rockets, destroy asteroids, and lose lives when hit by an asteroid. Ensure students understand the importance of correct code placement and sprite selection. Encourage them to test their game frequently to ensure it functions as expected.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop understanding of MakeCode Arcade for game creation.<br>2. Gain proficiency in creating and controlling game sprites.<br>3. Learn to implement game mechanics such as scoring and lives.<br>4. Understand how to detect and respond to sprite interactions.<br>5. Apply coding skills to create a complete Space Shooter game. | 1. Design and create a spaceship sprite in MakeCode Arcade.<br>2. Control the spaceship sprite using arrow keys and prevent it from going off the screen.<br>3. Set the number of lives for the spaceship.<br>4. Create and program asteroids to fly in from the right side of the screen.<br>5. Fire rockets from the spaceship when the A button is pressed. |

# Week 5

## Lesson: Platform Place

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through creating their first platform game using MakeCode Arcade. The lesson involves understanding the basics of platform games, creating a new project, designing a sprite, programming sprite movements, adding gravity, drawing a map with different elements, programming a jump function, testing the game, and adjusting the game's mechanics. Ensure students understand the code snippets and their purpose in the game's functionality. Encourage creativity in sprite and map design.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the basic concept and mechanics of platform games.<br>2. Create and design a sprite character in a game environment.<br>3. Implement movement controls for the sprite character.<br>4. Apply the concept of gravity in a game setting.<br>5. Design and create a game map with different elements such as ground, danger and goal tiles. | 1. Understand the concept of platform games and their mechanics.<br>2. Create a new project on arcade.makecode.com and design a sprite character.<br>3. Implement sprite movement controls using code.<br>4. Apply the concept of gravity to a sprite in a platform game.<br>5. Design a game map with ground, danger, and goal tiles.<br>6. Program a sprite to jump and move through the map.<br>7. Implement game mechanics such as danger tiles and a goal tile. |

# Week 6

## Lesson: Dino Jump

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

In this lesson, students will create an interactive game called 'Dino Jump' using MakeCode Arcade. They will learn how to draw a map, create a dino character, make it jump, add obstacles, keep score, and determine when the game is won. The lesson involves creating a new arcade project, drawing the map, creating the dino sprite, adding gravity and movement to the dino, making it jump, adding cactuses as obstacles, detecting collision with a tree, keeping score, and setting a win condition. The lesson concludes with a play and review session.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating an interactive game using MakeCode Arcade.<br>2. Understand how to draw a map, create a character, and add movement and gravity effects.<br>3. Learn to add obstacles and implement collision detection for game over scenarios.<br>4. Gain knowledge on how to keep score and determine winning conditions in a game.<br>5. Reflect on the game design process and the elements involved in creating an engaging game. | 1. Create a new arcade project on the MakeCode Arcade website.<br>2. Draw a map for the Dino Jump game, including ground tiles, walls, and a finish tile.<br>3. Create a dino sprite using the sprite editor and code.<br>4. Implement gravity and movement for the dino sprite, making it fall to the ground and move forwards through the map.<br>5. Program the A button to make the dino jump when it is touching the ground.<br>6. Add trees to the map as obstacles for the dino to jump over.<br>7. Implement game over functionality when the dino sprite hits a tree.<br>8. Keep score based on how long the player can go without hitting a tree.<br>9. Detect when the player reaches the end of the game and display a winning screen.<br>10. Play and review the Dino Jump game, reflecting on the elements of game design learned during the lesson. |

# Week 7

## Lesson: Monster Battle Arena

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through creating a 'Monster Battle Arena' game using MakeCode Arcade. They will learn to create player-controlled and AI-controlled sprites, implement combat mechanics, health systems, and AI behaviours. Students will also learn to create a new project, design sprites, make the monster move, implement a health system, create a combat system, and determine the winner. Encourage creativity and experimentation with the game's features.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop a player-controlled sprite and an AI-controlled monster in a game using MakeCode Arcade.<br>2. Create and manage a new project in MakeCode Arcade.<br>3. Implement a health system for player and monster sprites.<br>4. Develop a combat system where player and monster sprites can inflict damage on each other.<br>5. Implement a win/lose condition based on the health of player and monster sprites. | 1. Create and control a player sprite using MakeCode Arcade, ensuring it moves smoothly within the screen boundaries.<br>2. Program an AI-controlled monster sprite with randomized movement, simulating intelligent behavior.<br>3. Implement a health system that tracks and displays the health values of both the player and the monster during the game.<br>4. Develop a combat system that reduces player health upon collision with the monster and allows the player to shoot projectiles at the monster.<br>5. Determine the game's winner by programming conditions that end the game when either the player's or monster's health reaches zero. |

# Week 8

## Lesson: Donut Rush

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

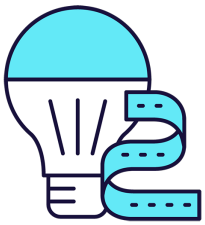
In this lesson, students will create an interactive game called 'Donut Rush' using MakeCode Arcade. They will learn to write code for creating game sprites, handling events like sprite overlaps, and controlling game logic. The lesson involves setting up the game, creating a new project, and defining variables to track the game's state. Students will also learn to create a function, set up the level, create the donuts, and start the game. They will add code to detect when the player sprite overlaps with a donut sprite and to check if the player has collected the target number of donuts. The lesson concludes with a wrap-up and play session.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop an understanding of game creation using MakeCode Arcade.<br>2. Learn to create and manage variables in a gaming context.<br>3. Understand the concept and application of functions in game development.<br>4. Gain skills in handling events such as sprite overlaps and controlling game logic.<br>5. Apply knowledge to create an interactive game with multiple levels and scoring system. | 1. Create a new project in MakeCode Arcade.<br>2. Set up the game by creating a splash screen, setting up variables, and creating a player sprite.<br>3. Create a function called 'startLevel' to organise the game's code.<br>4. Set up the level by adding code to the 'startLevel' function, including setting the background colour, displaying a level message, setting the target number of donuts to collect, and starting a countdown.<br>5. Create multiple donuts using a loop and place them randomly on the screen.<br>6. Start the game by calling the 'startLevel' function.<br>7. Collect donuts by detecting when the player sprite overlaps with a donut sprite, increasing the score, destroying the donut sprite, and playing a smile effect.<br>8. Complete the level by checking if the player has collected the target number of donuts, increasing the level, playing a 'jump up' sound, and starting a new level.<br>9. Wrap up the game and play it, aiming to collect as many donuts as possible within the time limit. |

# Module: Exploring Electronics and Light



This module explores the exciting world of electronics and light, using Microbit and LED strips. Teachers will guide students through creating colourful displays, sound-activated lights, visual thermometers, and even a precision game. The module encourages creativity, problem-solving, and teamwork, with students brainstorming and implementing their own Microbit projects. Teachers should ensure students understand each step and concept before progressing, and provide assistance during the project creation stages.

| **Duration** | **Equipment** |
|---|---|
| 8 weeks | Students can use any of these devices:<br>• Chromebook/Laptop/PC<br><br>Required Equipment:<br>• LED Strip with crocodile clips<br>• Microbit |
| **Module Goals** | **Module Outcomes** |
| 1. Understand and apply the principles of programming LED strips using Microbit projects.<br>2. Develop skills to create interactive LED displays that respond to sound and temperature changes.<br>3. Design and implement a game using LED strip and Microbit programming.<br>4. Enhance creativity and problem-solving skills through the design of LED flags and stacking effects.<br>5. Apply teamwork and project management skills in brainstorming and executing a group Microbit project. | 1. Programme a strip of LEDs to display colourful patterns using Microbit.<br>2. Design and implement an LED Strip Clapper that responds to sound, specifically a clap, to turn on and off.<br>3. Convert an LED strip into a visual thermometer that lights up and changes colour according to the current temperature.<br>4. Create a voice-activated 'Shooting Stars' display using an LED strip and Microbit.<br>5. Design and code tricolour flags using LED strips.<br>6. Create a stacking effect on an LED strip, controlled by Microbit, with the ability to increase and decrease the size of the stack.<br>7. Develop an LED Strip Precision Game that involves timing and accuracy.<br>8. Brainstorm, design, and implement a simple Microbit project in a team, demonstrating creativity and teamwork. |

# Week 1

## Lesson: Microbit LED Strip

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through programming a 30 LED strip using Microbits. Ensure understanding of creating a new Microbit project and adding the neopixel extension. Facilitate the setup of the LED strip and programming it to turn red. Assist with downloading the project onto the Microbit. Encourage creativity when programming the strip to show a rainbow of colours and rotating the rainbow. Finally, encourage exploration of other code blocks in the Neopixel toolbox.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply the process of programming a strip of LEDs using Microbits.<br>2. Develop skills in creating a new Microbit project and adding the necessary extensions.<br>3. Gain proficiency in setting up and programming the LED strip to display various colours.<br>4. Learn to download and implement the project on Microbits, observing the effects on the LED strip.<br>5. Explore and experiment with different code blocks in the Neopixel toolbox for creative lighting effects. | 1. Program a strip of 30 LEDs to light up in different ways using Microbits.<br>2. Create a new Microbit project and add the neopixel extension.<br>3. Set up the LED strip and interact with it using a variable.<br>4. Program the A button on the Microbit to turn all the LEDs red.<br>5. Program the LED strip to show a rainbow of colours when the Microbit turns on. |

# Week 2

## Lesson: LED Strip Clapper

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

In this lesson, students will create an LED Strip Clapper using a Microbit project. They will add the neopixel extension, set up the LED strip, and create an 'on' variable. The lesson will guide them to detect a clap, turning the LED strip on and off accordingly. They will download their code onto their microbit, connect it to the LED strip, and explore further improvements. Familiarity with Microbit and basic coding is beneficial.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating and managing a new Microbit project.<br>2. Understand and apply the neopixel extension for programming an LED strip.<br>3. Learn to set up and interact with the LED strip using variables.<br>4. Gain knowledge on creating and manipulating variables to control the state of the LED strip.<br>5. Develop the ability to detect sound inputs and use them to trigger changes in the LED strip's state. | 1. Develop a new Microbit project using makecode.microbit.org.<br>2. Integrate the neopixel extension into the project for LED strip programming.<br>3. Establish a variable for the LED strip and set its value to 30.<br>4. Create an 'on' variable to control the LED strip's state.<br>5. Implement a sound detection feature to trigger the LED strip's state change. |

# Week 3

## Lesson: Microbit LED Strip Thermometer

| | | | | |
|---|---|---|---|---|
| Beginner | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare for this lesson by familiarising yourself with the Microbit project platform and the neopixel extension. Understand how to set up the LED strip and how to program the A button to display temperature. Be ready to guide students in lighting up the LED lights according to temperature readings and downloading their projects onto their Microbits. Ensure you know how to correctly connect the LED strip to the Microbit.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills to create and manage a new Microbit project.<br>2. Understand and apply the neopixel extension to program the LED strip.<br>3. Gain knowledge on setting up the LED strip and displaying temperature on the Microbit screen.<br>4. Learn to light up the LED lights on the strip according to the temperature readings.<br>5. Acquire practical skills in downloading the project, connecting the LED strip to the Microbit, and testing the functionality. | 1. Create a new Microbit project using makecode.microbit.org.<br>2. Add the neopixel extension to the project for LED strip programming.<br>3. Set up the LED strip in the project with a value of 30, representing the 30 LEDs on the strip.<br>4. Program the A button to display the temperature on the Microbit screen.<br>5. Display the temperature by lighting up the LED lights on the strip, with the number of lights corresponding to the temperature reading.<br>6. Download the project and transfer it to the Microbit.<br>7. Connect the LED strip to the Microbit using the specified pin connections and power it using a USB cable. |

# Week 4

## Lesson: Shooting Stars

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students in creating a Microbit project, adding the neopixel extension, and setting up the LED strip. Facilitate the creation of a 'star' that lights up with a loud sound, and ensure students can test this on their LED strip. Assist students in making the 'star' shoot along the strip and adding random colours. Finally, ensure students can download and test their code, encouraging them to create multiple shooting stars.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating and managing a new Microbit project.<br>2. Understand and apply the neopixel extension to program the LED strip.<br>3. Gain proficiency in setting up and programming the LED strip using code blocks.<br>4. Learn to utilise the microphone in the microbit to detect sound and trigger LED actions.<br>5. Acquire knowledge on how to test and debug the project on the LED strip.<br>6. Master the concept of pixel shifting to create the illusion of moving light.<br>7. Experiment with random colour generation for the LED strip.<br>8. Learn to download and implement the code onto the microbit for real-world testing. | 1. Create and manage a new Microbit project.<br>2. Integrate the neopixel extension into the project.<br>3. Set up and programme the LED strip using the provided code.<br>4. Develop a function to light up the first LED on the strip white when a loud sound is detected.<br>5. Test the function on the LED strip and ensure it works as expected.<br>6. Implement a function to make the 'star' shoot along the strip.<br>7. Enhance the function to display stars in random colours.<br>8. Download and test the final code on the microbit, ensuring different colour 'stars' shoot along the strip when a loud noise is made. |

# Week 5

## Lesson: LED Flags

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students in creating LED flags using a Microbit project. They will need to understand how to add the neopixel extension and set up the LED strip. Facilitate as they create bicolor and tricolor flags, using the example of Malta and Ireland respectively. Encourage creativity and problem-solving skills for the challenge of representing the American flag.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply the concept of bicolor and tricolor flags using LED strips.<br>2. Create and manage a new Microbit project effectively.<br>3. Utilise the neopixel extension to program the LED strip.<br>4. Develop skills to set up and interact with the LED strip using code.<br>5. Apply coding skills to create complex patterns, such as the American flag, on the LED strip. | 1. Construct bicolor and tricolor flags using LED strips.<br>2. Utilise the neopixel extension to program the LED strip.<br>3. Set up and interact with the LED strip using a variable.<br>4. Apply the concept of ranges to light up specific sections of the LED strip.<br>5. Code the LED strip to represent complex flag designs, such as the American flag. |

# Week 6

## Lesson: LED Stacking

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare for this lesson by familiarising yourself with the Microbit project platform and the neopixel extension. Understand how to set up an LED strip and create variables to store the strip and the amount of LEDs. Be ready to guide students in creating a function to show the LED stack, and programming buttons to increase and decrease the stack. Ensure students know how to download their code and connect their LED strip to their microbit.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop skills in creating and managing a new Microbit project.<br>2. Understand and apply the neopixel extension for LED programming.<br>3. Learn to set up and interact with the LED strip using variables.<br>4. Develop competency in creating and using functions to control LED display.<br>5. Gain experience in programming button controls to manipulate LED stack size. | 1. Create a new Microbit project using makecode.microbit.org.<br>2. Add the neopixel extension to the project for LED strip programming.<br>3. Set up the LED strip with a variable storing the strip, set to a value of 30.<br>4. Create an 'amount' variable to store the number of LEDs in the stack.<br>5. Develop a 'showStack' function to display the stack of lit LEDs.<br>6. Create a range of LEDs on the strip to light up, using the 'amount' variable, and call the 'showStack' function from the 'on start' block.<br>7. Program button A to increase the LED stack by adding 1 to the 'amount' variable and calling the 'showStack' function.<br>8. Program button B to decrease the LED stack by subtracting 1 from the 'amount' variable and calling the 'showStack' function.<br>9. Download the code onto a microbit, connect the LED strip using crocodile clips, and test the LED stack's increase and decrease functions with buttons A and B. |

# Week 7

## Lesson: LED Strip Precision Game

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through creating an interactive LED strip game using a Microbit project. Familiarise yourself with the neopixel extension and the process of setting up the LED strip. Understand the purpose of the four variables: 'target', 'position', 'delay', and 'increment'. Be ready to explain how to set up the level, create a refresh function, and make the blue light move. Prepare to guide students through the steps of going back to the start, hitting the target, and handling a missed target. Finally, ensure you can assist students in downloading their code and playing the game.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit
- LED Strip with crocodile clips

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand and apply the concept of LED strip programming using Microbit.<br>2. Develop skills in creating and manipulating variables in a coding project.<br>3. Learn to create and use functions for specific tasks within a coding project.<br>4. Gain proficiency in using conditional statements to control game outcomes.<br>5. Develop the ability to download and test code on a physical device. | 1. Program an LED strip to light up specific LEDs in response to user input.<br>2. Create and manipulate variables to control game mechanics in a Microbit project.<br>3. Implement the neopixel extension to interact with an LED strip.<br>4. Design a function to refresh LED lights based on variable values.<br>5. Download and test the code on a physical Microbit device. |

# Week 8

## Lesson: Microbit Lab

| | | |
|---|---|---|
| Advanced | 60 mins | Teacher led |

Prepare to introduce the concept of Microbit projects, demonstrating a simple LED pattern to inspire creativity. Organise students into small groups for brainstorming, emphasising teamwork and achievable project ideas. Facilitate a feedback session after idea presentations, guiding project simplification if necessary. Assist during project creation, encouraging peer support and discovery sharing. Finally, conduct a 'Show and Tell' session, celebrating student effort and creativity, reinforcing learning objectives and the importance of teamwork.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC

Required equipment for this lesson:

- Microbit

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop creative and achievable project ideas using basic Microbit blocks.<br>2. Collaborate effectively in small groups to brainstorm, plan and execute a Microbit project.<br>3. Present project ideas clearly and receive feedback constructively.<br>4. Apply problem-solving skills to create a Microbit project based on the brainstormed idea.<br>5. Reflect on the project creation process, discussing changes made, challenges faced, and skills learned. | 1. Brainstorm and develop a simple Microbit project idea in a group setting.<br>2. Present the project idea to the class, explaining the planned LED patterns and inputs.<br>3. Receive, incorporate, and respond to feedback on the project idea.<br>4. Create a Microbit project based on the brainstormed idea, using basic Microbit blocks.<br>5. Present the final Microbit project to the class, explaining the coding process and any changes made during the project creation. |

# Module: Discovering Artificial Intelligence

This module explores the fascinating world of artificial intelligence (AI), starting with an introduction to AI models, their types, applications, and limitations. Students will gain hands-on experience creating image and pose models using Google's Teachable Machine, and applying these models in interactive games using Scratch. The module culminates in a project where students conceptualise, plan, and build their own AI Scratch project, applying their newfound knowledge and skills. Teachers should familiarise themselves with the tools and concepts, and be prepared to guide students through each step, encouraging creativity and problem-solving throughout.

| **Duration** | **Equipment** |
| --- | --- |
| 4 weeks | Students can use any of these devices:<br>• Chromebook/Laptop/PC<br>• iPad/Tablet<br><br>Required Equipment:<br>• Webcam/camera |
| **Module Goals** | **Module Outcomes** |
| 1. Understand the fundamentals of AI models, their types, applications, limitations, and ethical considerations.<br>2. Develop an image model using Google's Teachable Machine and apply it in a practical project.<br>3. Create an interactive game using Scratch and Google Teachable Machine, incorporating elements of randomisation and conditionals.<br>4. Design and develop a pose model using Google's Teachable Machine, and apply it in a space game project.<br>5. Conceptualise, plan, and execute an original AI Scratch project, demonstrating creativity, problem-solving, and application of AI knowledge. | 1. Understand and explain the function, types, applications, and limitations of AI models, including ethical considerations.<br>2. Create an image model using Google's Teachable Machine for a rock, paper, scissors game.<br>3. Develop a Rock, Paper, Scissors game using Scratch and Google Teachable Machine, incorporating variables, randomisation, and conditionals.<br>4. Create a pose model using Google's Teachable Machine for a space game, understanding the importance of testing and adjusting the model.<br>5. Conceptualise, plan, and build a unique AI Scratch project, demonstrating creativity, problem-solving, and the ability to seek and incorporate feedback. |

# Week 1

## Lesson: An Introduction to AI Models

| | | | |
|---|---|---|---|
| Beginner | 20 mins | Teacher/Student led | Student Quiz |

Prepare to introduce students to AI models, explaining their function and various types. Discuss different learning methods such as supervised, unsupervised, and reinforcement learning. Explore the diverse applications of AI models, from speech recognition to autonomous vehicles. Discuss the limitations of AI models, including data quality and computational resources. Finally, delve into the ethics of AI models, discussing responsibility, privacy, transparency, and fairness.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Understand the concept and purpose of AI models.<br>2. Identify different types of AI models and their learning methods.<br>3. Recognise various applications of AI models in real-world scenarios.<br>4. Appreciate the limitations and challenges associated with AI models.<br>5. Reflect on the ethical considerations in the use of AI models. | 1. Identify and describe the different types of AI models: Supervised Learning, Unsupervised Learning, and Reinforcement Learning.<br>2. Explain the various applications of AI models, including speech recognition, image recognition, natural language processing, recommendation systems, and autonomous vehicles.<br>3. Discuss the limitations of AI models, focusing on data quality, computational resources, transparency, privacy, and security.<br>4. Understand the ethical considerations related to AI models, including responsibility, privacy, transparency, and fairness.<br>5. Demonstrate a basic understanding of how AI models function, their uses, limitations, and ethical implications. |

## Lesson: Create an Image Model

| | | | |
|---|---|---|---|
| Intermediate | 20 mins | Teacher/Student led | Student Quiz |

Familiarise yourself with Google's Teachable Machine tool before the lesson. Ensure students understand the concept of machine learning and how it applies to image recognition. Encourage students to take clear images for their classes and emphasise the importance of quality over quantity. Guide them through the process of training, testing, and exporting their models. Reinforce the practical application of these skills in future projects.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

| Learning Goals | Learning Outcomes |
| --- | --- |
| 1. Understand and utilise Google's Teachable Machine to create an image model.<br>2. Create and define classes within an image model project.<br>3. Add and manage image samples to each class for effective model training.<br>4. Train, test, and refine the image model to ensure accurate gesture recognition.<br>5. Export and save the created image model for future use in projects. | 1. Utilise Google's Teachable Machine to create an image model.<br>2. Create and categorise classes within an image model project.<br>3. Add and record images to each class using a webcam.<br>4. Train the image model using the added images and understand the process of machine learning.<br>5. Test the model's performance, make necessary adjustments, and export the model for future use. |

# Week 2

## Lesson: Scratch AI Rock, Paper, Scissors Game

| | | | | |
|---|---|---|---|---|
| Intermediate | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students through creating a Rock, Paper, Scissors game using Scratch and Google Teachable Machine. Ensure they understand the use of variables, randomisation, and conditionals. They'll need to set up Scratch and TM2Scratch, add a sprite, create variables, and load a Teachable Machine Model. They'll also learn to set a confidence threshold, get player choice, determine game outcomes, and add enhancements. Encourage creativity and problem-solving throughout.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop a Rock, Paper, Scissors game using Scratch and Google Teachable Machine.<br>2. Understand and apply the use of variables in Scratch for storing player's choice, computer's choice, and the result of the game.<br>3. Implement randomisation in Scratch to simulate the computer's choice in the game.<br>4. Integrate Google Teachable Machine Image models in Scratch projects for gesture recognition.<br>5. Understand and adjust the confidence threshold for AI model to improve accuracy of gesture recognition. | 1. Develop a Rock, Paper, Scissors game using Scratch and Google Teachable Machine.<br>2. Set up Scratch and TM2Scratch for the game development.<br>3. Create and utilise variables to store player's choice, computer's choice, and the game result.<br>4. Implement randomisation for computer's choice in the game.<br>5. Load and use a Teachable Machine Image model for hand gesture recognition.<br>6. Set and adjust the confidence threshold for the AI model.<br>7. Recognise and interpret player's choice through hand gestures.<br>8. Develop game logic to determine the game result: draw, win, or lose.<br>9. Improve the game by enhancing the image model, adding new features like sound effects, and improving user interaction. |

# Week 3

## Lesson: Create a Pose Model

| | | | |
|---|---|---|---|
| Intermediate | 20 mins | Teacher/Student led | Student Quiz |

Prepare to guide students through creating a pose model using Google's Teachable Machine. Familiarise yourself with the tool and the process of creating classes, adding images, and training the model. Be ready to troubleshoot any issues with webcam permissions or image quality. Ensure students understand the importance of testing their model and making necessary adjustments. Finally, assist them in exporting their model for future use in projects like an AI-powered space game.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop an understanding of Google's Teachable Machine and its application in creating pose models.<br>2. Acquire skills to create and categorise classes within a pose model.<br>3. Learn to add and manage image samples for each class to train the model.<br>4. Gain proficiency in training and testing the model for different poses.<br>5. Master the process of exporting the model for future use in other projects. | 1. Operate Google's Teachable Machine to create a pose model.<br>2. Define and create classes for the pose model.<br>3. Add and categorise images into the respective classes: Tilt Left, Tilt Right, and No Tilt.<br>4. Train the pose model using the categorised images and test its performance.<br>5. Export the created pose model and obtain a shareable link for future use. |

## Lesson: Scratch AI Pose Space Game

| | | | | |
|---|---|---|---|---|
| Advanced | 60 mins | Teacher/Student led | Student Quiz | Student Challenge |

Prepare to guide students in creating a Scratch AI Pose Space Game. They'll learn to use Scratch and Google Teachable Machine to control a spaceship with tilt poses. They'll set up Scratch, add a rocketship sprite, load a Teachable Machine Model, display pose labels, set a confidence threshold, and make the spaceship move. They'll also add a star sprite, make stars fall, and face a challenge to improve their game. Ensure students understand each step, and encourage creativity in the challenge.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

| Learning Goals | Learning Outcomes |
| --- | --- |
| 1. Develop skills in using Scratch and Google Teachable Machine to create a game.<br>2. Understand how to control a sprite using pose models.<br>3. Learn to set up and adjust the confidence threshold for an AI model.<br>4. Gain knowledge on how to create and manipulate clones of sprites in Scratch.<br>5. Apply creativity to enhance and personalise the game with additional features. | 1. Create a Space game using Scratch and Google Teachable Machine.<br>2. Set up Scratch and TMPose2Scratch for a new project.<br>3. Integrate a Teachable Machine Pose model into the Scratch project.<br>4. Control a sprite's movement using pose labels and confidence thresholds.<br>5. Enhance the game by adding falling sprites and scoring mechanisms. |

# Week 4

## Lesson: Crafting Your Own AI Project

| | | |
|---|---|---|
| Advanced | 60 mins | Teacher/Student led |

In this lesson, students will utilise their knowledge of AI and Scratch to create their own AI project. They will brainstorm ideas, focusing on real-life routines or challenges that could be enhanced with AI. After selecting their favourite idea, they will create a project proposal, seek feedback, refine their idea, and plan their project. They will then prototype and code their project, before presenting and demonstrating their work. Finally, they will reflect on their learning journey and the process of creating their project.

Students can use any of these devices (and can share if necessary):

- Chromebook/Laptop/PC
- iPad/Tablet

Required equipment for this lesson:

- Webcam/camera

| Learning Goals | Learning Outcomes |
|---|---|
| 1. Develop the ability to conceptualise and plan an AI project.<br>2. Enhance brainstorming skills and generate creative AI project ideas.<br>3. Gain proficiency in creating and refining a project proposal.<br>4. Acquire skills in prototyping and coding an AI project using Scratch and Google Teachable Machine.<br>5. Improve presentation skills and ability to reflect on the learning process and project outcomes. | 1. Generate and evaluate 3-5 AI project ideas, drawing inspiration from daily routines or challenges.<br>2. Formulate a detailed project proposal, including project name, purpose, required features, and necessary components.<br>3. Seek and incorporate feedback from peers or teachers to refine the project idea and plan.<br>4. Code, prototype, and test the core features of the AI project, using problem-solving skills to overcome any issues.<br>5. Present and demonstrate the final AI project, reflecting on the challenges faced, solutions found, and lessons learned. |