



CODING  
IRELAND

# Teacher Learning Plan

Digital Skills  
Curriculum 2024/25

Transition Year

# Table of Contents



- [How to Use This Learning Plan](#)
- [Module: Introduction to Coding Concepts](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
  - [Week 8](#)
  - [Week 9](#)
- [Module: Exploring Microbit Programming](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
  - [Week 8](#)
  - [Week 9](#)
- [Module: Game Design Essentials](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
  - [Week 8](#)
- [Module: Robotic Cars and Automation](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
  - [Week 8](#)
  - [Week 9](#)
- [Module: Exploring Digital Art and Design](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
  - [Week 8](#)
- [Module: Web Design Basics](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
  - [Week 8](#)
- [Module: Dynamic Web Design](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
- [Module: Introduction to Python](#)
  - [Week 1](#)
  - [Week 2](#)
  - [Week 3](#)
  - [Week 4](#)
  - [Week 5](#)
  - [Week 6](#)
  - [Week 7](#)
  - [Week 8](#)
  - [Week 9](#)
  - [Week 10](#)

# How to Use This Learning Plan

This learning plan provides an overview of all the modules available for Transition 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](mailto:info@codingireland.ie) or 01 584 9955.

# Module: Introduction to Coding Concepts







This module introduces students to the world of coding, starting with a basic understanding of what coding is and its importance in today's world. The module progresses to practical coding exercises using Scratch, where students will learn to create games, animations, and projects. They will learn to control sprites, use sensing blocks, and create backdrops. The module also covers programming keys on a keyboard, creating sprites with multiple costumes, introducing randomness to elements of a project, and creating unique patterns. The module concludes with a project showcase. Teachers should facilitate exploration, answer questions, and encourage creativity and practice.

Duration	Equipment
11 weeks	Students can use any of these devices: <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> <li>• iPad/Tablet</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Grasp fundamental coding concepts and their applications in various fields.</li> <li>2. Develop proficiency in using Scratch to create games, animations, and projects.</li> <li>3. Understand and apply coding principles to create interactive games and simulations.</li> <li>4. Learn to manipulate variables, loops, and sprites to create complex patterns and autonomous systems.</li> <li>5. Develop the ability to showcase and explain personal coding projects effectively.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the basic concepts of coding and its applications in the real world.</li> <li>2. Navigate and utilise Scratch to create simple games and animations, including manipulating sprites, blocks, loops, and backdrops.</li> <li>3. Develop interactive games using Scratch, incorporating elements such as moving sprites, backdrops, sensing blocks, and keyboard controls.</li> <li>4. Apply advanced Scratch features to create complex animations and games, including cloning, colour detection, and autonomous navigation.</li> <li>5. Present and showcase a completed coding project, demonstrating a comprehensive understanding of coding concepts and practical application in Scratch.</li> </ol>

# Week 1

## Lesson: An Introduction to Coding

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






For this lesson, teachers should familiarise themselves with the basic concepts of coding, its importance in today's digital age, and the various coding languages. They should be prepared to explain how coding is the backbone of modern technology, a valuable skill in the job market, and a tool for enhancing problem-solving skills. Teachers should also be ready to introduce Scratch, a beginner-friendly coding language, and discuss the prevalence of coding in everyday devices and applications.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the concept of coding and its role as a language for digital devices.</li> <li>2. Appreciate the importance of coding in modern technology, job market, and its contribution to problem-solving and logical thinking.</li> <li>3. Identify different coding languages and their applications, with a focus on Scratch as a beginner-friendly option.</li> <li>4. Recognise the prevalence of coding in everyday devices and applications, and its broad implications in daily life.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the concept of coding and its role in instructing digital devices.</li> <li>2. Appreciate the importance of coding in creating modern technology, enhancing job prospects, and improving problem-solving skills.</li> <li>3. Identify Scratch as a beginner-friendly coding language and recognise its role in creating stories, games, and animations.</li> <li>4. Recognise the presence and influence of coding in everyday devices and applications.</li> <li>5. Develop curiosity and interest in learning more about coding and its broad applications.</li> </ol>

## Lesson: Scratch Tutorial

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

This lesson introduces students to Scratch, a coding platform for creating games and animations. Teachers should familiarise themselves with the Scratch website and its functionalities. The lesson guides students through creating a project, removing the default sprite, adding a new sprite, making it move, adjusting values, creating a loop, adding a backdrop, and encourages further exploration. Teachers should be prepared to assist with any technical difficulties and encourage experimentation.

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

- Chromebook/Laptop/PC
- iPad/Tablet

## Learning Goals

1. Understand and navigate the Scratch coding platform.
2. Manipulate sprites by adding, removing, and controlling their movements.
3. Apply basic coding concepts such as loops and event triggers.
4. Modify code blocks to alter sprite behaviour.
5. Explore and experiment with various Scratch functionalities to create unique projects.

## Learning Outcomes

1. Identify Scratch as a coding platform for creating games, animations and projects.
2. Navigate and utilise the Scratch website interface.
3. Remove default sprites and add new ones from the sprite library.
4. Implement basic coding blocks to manipulate sprite movement.
5. Modify values within code blocks to alter sprite behaviour.
6. Create a loop within the code to repeat specific actions.
7. Add a backdrop from the library to enhance the visual aspect of the project.
8. Explore and experiment with various code blocks to diversify sprite actions.

## Week 2

### Lesson: Paddle Ball Game

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

Prepare to guide students through creating a Paddle Ball Game using Scratch. They'll learn to move sprites, change backdrops, and use sensing blocks. They'll create a new Scratch project, add a paddle and a football sprite, position the ball, make it bounce, control the paddle, make the ball bounce off the paddle, add a backdrop, add a game over line and program the game over. Ensure students understand X and Y coordinates, and how to use the Scratch coding blocks.



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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop skills in using Scratch to create a simple game.</li> <li>2. Understand and apply the concept of sprites and backdrops in Scratch.</li> <li>3. Learn to control sprite movements using mouse input.</li> <li>4. Implement game logic using conditional statements in Scratch.</li> <li>5. Understand and apply the concept of X and Y coordinates to position sprites.</li> </ol>	<ol style="list-style-type: none"> <li>1. Manipulate sprites and backdrops in Scratch.</li> <li>2. Utilise X and Y coordinates to position sprites.</li> <li>3. Implement code to control sprite movement and interaction.</li> <li>4. Use sensing blocks to detect sprite collision and mouse position.</li> <li>5. Create a game over condition using colour detection.</li> </ol>

## Week 3

### Lesson: Racing Car

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

This lesson involves creating a racing car game using Scratch. Teachers should familiarise themselves with Scratch and its features. The lesson starts with creating a new Scratch project and adding a car sprite. The car is then resized and a racing track is drawn. The car is placed on the track and a speed variable is created. The car's location is detected and it is programmed to move forwards, backwards, left, and right. Finally, students test drive their car. Teachers should ensure students understand each step before moving on.

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





- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop skills in creating and managing a new Scratch project.</li> <li>2. Understand and apply the process of adding and modifying sprites in Scratch.</li> <li>3. Gain knowledge in drawing and editing backdrops in Scratch.</li> <li>4. Learn to create and manipulate variables to control sprite properties.</li> <li>5. Acquire skills in programming sprite movements and interactions using Scratch blocks.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop a new Scratch project and remove the default sprite.</li> <li>2. Upload a provided car sprite into the Scratch project.</li> <li>3. Resize the car sprite to 10% of its original size using Scratch code.</li> <li>4. Draw a racing track using the backdrop editor in Scratch.</li> <li>5. Position the car sprite on the track and code its starting position and direction.</li> <li>6. Create a 'speed' variable to control the car's speed.</li> <li>7. Program the car to detect its location and adjust its speed accordingly.</li> <li>8. Code the car to move forwards and backwards using the up and down arrow keys.</li> <li>9. Code the car to turn left and right using the left and right arrow keys.</li> <li>10. Test drive the car using the programmed controls and observe its speed changes on different surfaces.</li> </ol>



## Week 4

### Lesson: Red v Green v Blue

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

Prepare to guide students in creating a new Scratch project, focusing on creating and manipulating sprites. Students will create a red dot sprite, then duplicate and recolour it to create green and blue versions. They will then learn to create a 'count' variable and use it to clone the dot sprite 100 times. The clones will be coded to appear randomly on the screen, move around, and 'infect' each other, changing colours according to a set rule. Encourage students to think of ways to improve the project.






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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop skills in creating and managing a new Scratch project.</li> <li>2. Acquire knowledge in creating and modifying sprites and costumes in Scratch.</li> <li>3. Understand and apply the concept of variables in Scratch programming.</li> <li>4. Learn to code sprite clones and manage their behaviour in Scratch.</li> <li>5. Enhance problem-solving skills by improving and customising the project.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop a new Scratch project and remove the default sprite.</li> <li>2. Create a red dot sprite using the sprite editor.</li> <li>3. Generate green and blue costumes for the sprite.</li> <li>4. Create a 'count' variable for counting up to 100.</li> <li>5. Code the sprite to clone itself 100 times using the 'count' variable.</li> <li>6. Programme each clone to randomly select a costume and position, and then appear on the screen.</li> <li>7. Code the dots to move in a random direction and bounce off the screen edges.</li> <li>8. Programme the dots to change colour when they touch another dot, according to specific rules.</li> <li>9. Improve the project based on personal ideas and creativity.</li> </ol>

## Week 5

### Lesson: Pattern Creator

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

Prepare to guide students through an engaging exploration of pattern creation using Scratch. Familiarise yourself with the Scratch interface and pen tool, as well as the process of creating a new project and adding sprites. Be ready to explain the use of variables, loops, and how to manipulate pen colour and size. Encourage students to experiment with different degrees and pen sizes to create unique patterns. Wrap up by reinforcing the importance of practice and creativity in mastering coding.





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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Master the use of Scratch for pattern creation.</li> <li>2. Understand and apply the use of variables in creating complex shapes and patterns.</li> <li>3. Manipulate the pen tool to draw and create unique patterns.</li> <li>4. Experiment with different degrees and pen sizes to alter pattern outcomes.</li> <li>5. Apply creativity in coding to produce vibrant and unique patterns.</li> </ol>	<ol style="list-style-type: none"> <li>1. Code a Scratch project to create basic patterns using the pen tool.</li> <li>2. Implement the use of variables to manipulate pattern creation.</li> <li>3. Adjust pen colour and size to enhance pattern design.</li> <li>4. Utilise loops and conditional statements to control pattern formation.</li> <li>5. Experiment with different variable values to create unique patterns.</li> </ol>

## Week 6

### Lesson: Attack of the Dots

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

Prepare for an interactive lesson where students will create a game using Scratch. They will learn to control a coloured disc, clone attacking dots, and detect the colour of the dots. Ensure students understand how to remix a starter project, make the disc spin, clone the ball, prevent the ball from appearing too close to the disc, make the ball move, detect the colour of the ball, create purple and orange balls, and change the code for the purple and orange balls. Wrap up by congratulating students on their newly acquired skills.






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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop skills in using Scratch to create an interactive game.</li> <li>2. Understand how to control a coloured disc using keyboard inputs.</li> <li>3. Learn to clone game elements and set their behaviour.</li> <li>4. Master the technique of colour detection for game mechanics.</li> <li>5. Apply problem-solving skills to prevent game elements from spawning too close to the player.</li> </ol>	<ol style="list-style-type: none"> <li>1. Master the use of Scratch to create an interactive game.</li> <li>2. Control a coloured disc using keyboard inputs.</li> <li>3. Clone and manipulate game elements, such as coloured dots, using Scratch code.</li> <li>4. Implement colour detection to trigger game events.</li> <li>5. Modify and customise game elements to enhance gameplay.</li> </ol>

## Week 7

### Lesson: Autonomous Car

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

Prepare to guide students through the process of understanding how autonomous cars work. Facilitate the creation of a Scratch project where students will program their own autonomous car, incorporating elements such as car sprites, speed variables, and sensor-driven navigation. Encourage students to experiment with different track designs and speeds, fostering a deeper understanding of autonomous vehicle technology.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the concept and workings of an autonomous car.</li> <li>2. Develop skills in creating a new Scratch project and manipulating sprites.</li> <li>3. Learn to use variables and conditional statements in Scratch to control sprite movements.</li> <li>4. Apply knowledge of sensors in programming an autonomous car to navigate a track.</li> <li>5. Enhance problem-solving skills by implementing speed control and reverse functions in the autonomous car project.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the functioning of an autonomous car and its use of sensors for navigation.</li> <li>2. Create a new Scratch project and manipulate sprites and backdrops.</li> <li>3. Program the car to move and navigate using colour detection and conditional statements.</li> <li>4. Control the speed of the car using variables and keyboard inputs.</li> <li>5. Implement a reverse function to correct the car's course when it deviates from the track.</li> </ol>

## Week 8

### Lesson: Rocket Lander

● Advanced	🕒 60 mins	👥 Teacher/Student led	📝 Student Quiz	💡 Student Challenge
------------	-----------	-----------------------	----------------	---------------------

Prepare to guide students through creating a rocket landing game using Scratch. The lesson involves programming gravity, controlling rocket movement, creating animations for rocket thrust and explosion, and adding a fuel limit for an extra challenge. Ensure students understand the concept of variables and conditions in coding. Encourage creativity and problem-solving as they experiment with their game.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the concept of vertical rocket landing and its challenges.</li> <li>2. Develop a game using Scratch, simulating a rocket landing scenario.</li> <li>3. Implement gravity and movement controls in the game using code blocks.</li> <li>4. Create and use costumes to animate rocket thrust and explosion.</li> <li>5. Introduce and manage a fuel limit for added complexity in the game.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and explain the functionality of the Space X Falcon 9 rocket.</li> <li>2. Create a basic game in Scratch, including setting up a starter project.</li> <li>3. Program gravity and booster functions for a rocket sprite in Scratch.</li> <li>4. Design and implement visual effects such as rocket thrust and explosion in Scratch.</li> <li>5. Implement controls for rocket movement and landing, including fuel limits and landing conditions.</li> </ol>

## Week 9

### Lesson: Project Showcase

● Advanced

🕒 90 mins

👥 Teacher/Student led

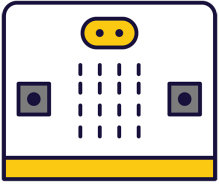
For the 'Project Showcase' lesson, teachers should encourage students to brainstorm creatively and select a project idea they are passionate about. Facilitate the planning phase, ensuring students consider the design and functionality of their projects. Support students during the coding process, reminding them to refer to previous lessons if needed. Encourage testing and debugging as part of the learning process. Finally, create a supportive environment for students to present their projects, highlighting their achievements and challenges faced.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Generate and select creative project ideas.</li> <li>2. Plan and design a project, identifying necessary resources and steps.</li> <li>3. Apply coding skills to create a unique project.</li> <li>4. Test, debug, and refine the project to ensure functionality.</li> <li>5. Present the final project, explaining its workings and discussing challenges faced.</li> </ol>	<ol style="list-style-type: none"> <li>1. Generate and select a creative project idea.</li> <li>2. Formulate a detailed plan for the chosen project, including visual design and required coding blocks.</li> <li>3. Apply coding skills to execute the project plan, using previous lessons as reference.</li> <li>4. Perform testing and debugging to ensure the project functions as intended.</li> <li>5. Present the final project effectively, explaining its workings and discussing faced challenges.</li> </ol>

# Module: Exploring Microbit Programming







This module will guide teachers through introducing students to the world of microbit programming. Teachers will facilitate students in creating various projects, from a simple step counter to a voting system. Each lesson provides hands-on experience with coding, fostering students' problem-solving and critical thinking skills. Teachers should be prepared to assist with coding and troubleshooting, and encourage students to experiment and explore the capabilities of microbits. The module culminates in a showcase, allowing students to present their projects and reflect on their learning journey.

Duration	Equipment
9 weeks	Students can use any of these devices: <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> </ul> Required Equipment: <ul style="list-style-type: none"> <li>• Microbit</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Master the basics of Microbit programming, including creating projects, writing and deleting code, and connecting Microbits to computers.</li> <li>2. Develop practical applications of Microbit programming, such as creating a step counter, a reaction timer game, and a guessing game.</li> <li>3. Understand and utilise the built-in features of Microbits, such as the accelerometer, magnetometer, and temperature sensor.</li> <li>4. Apply Microbit programming skills to create interactive games and systems, including a paddle ball game, a voting system, and a 'Chase the Dot' game.</li> <li>5. Present and showcase individual Microbit projects, demonstrating a comprehensive understanding of Microbit programming and its applications.</li> </ol>	<ol style="list-style-type: none"> <li>1. Program a microbit to display messages, react to button presses, show icons, play melodies, and respond to movement.</li> <li>2. Develop a step counter using the microbit's accelerometer, displaying the number of steps taken.</li> <li>3. Create a reaction timer game using a microbit, measuring reaction times to random visual prompts.</li> <li>4. Design a 'Higher or Lower' game on a microbit, programming button inputs and implementing game logic.</li> <li>5. Control a Scratch Paddle Ball game using a microbit, manipulating the paddle by tilting the microbit.</li> <li>6. Transform a microbit into a compass and thermometer, programming the buttons to use the built-in sensors.</li> <li>7. Establish a voting system using multiple microbits, programming them to cast votes, tally the votes, and reset the system.</li> <li>8. Develop a 'Chase the Dot' game on a microbit, defining variables, creating and calling functions, and using gestures to control movement.</li> <li>9. Present a showcase of the microbit projects developed throughout the course, demonstrating proficiency in microbit programming.</li> </ol>

# 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
<ol style="list-style-type: none"> <li>1. Understand the basic functionality and features of a microbit.</li> <li>2. Create a new project using the MakeCode for microbit website.</li> <li>3. Use the Project Editor to write and simulate code.</li> <li>4. Program the microbit to display numbers and text on its LED grid.</li> <li>5. Program the microbit to respond to button presses with specific actions.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify the functions and capabilities of a microbit.</li> <li>2. Create a new project on the MakeCode for microbit website.</li> <li>3. Understand the layout and functions of the Project Editor.</li> <li>4. Write and execute code to display numbers and names on the microbit.</li> <li>5. Program the microbit to respond to button presses with specific displays.</li> <li>6. Connect and download code to an actual microbit device.</li> <li>7. Compose and program a melody to play on the microbit.</li> <li>8. Explore and experiment with different coding blocks and functions.</li> </ol>



## Week 2

### Lesson: Microbit Step Counter

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

Prepare to guide students through creating a Microbit step counter. They'll start a new project on [makecode.microbit.org](https://makecode.microbit.org), create and set up a 'steps' variable, and use the accelerometer to detect steps. They'll write code to display the step count and send it to their Microbit. After connecting a power source, they'll secure the Microbit to their person and start walking. They'll adjust the code to count every step and resend the updated code to their Microbit. Caution them to be careful while walking with 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
<ol style="list-style-type: none"> <li>1. Develop a basic understanding of Microbit programming and project creation.</li> <li>2. Learn to create and set up variables in Microbit.</li> <li>3. Understand the use of accelerometer sensor in Microbit for step detection.</li> <li>4. Gain skills to display data on Microbit using LEDs.</li> <li>5. Learn to modify and resend code to Microbit for improved functionality.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop a new Microbit project using the <a href="https://makecode.microbit.org">makecode.microbit.org</a> website.</li> <li>2. Create and set up a 'steps' variable to record the number of steps taken.</li> <li>3. Utilise the accelerometer sensor in Microbits to detect and record steps.</li> <li>4. Display the recorded number of steps on the Microbit using its LEDs.</li> <li>5. Modify the code to accurately count every step taken, and resend the updated code to the Microbit.</li> </ol>

## Week 3

### 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
<ol style="list-style-type: none"> <li>1. Develop skills in creating and managing a new project on the Micro:bit platform.</li> <li>2. Acquire knowledge on how to create and display messages using code.</li> <li>3. Understand and apply the concept of countdowns and delays in programming.</li> <li>4. Learn to create and utilise variables for storing time stamps.</li> <li>5. Gain proficiency in recording and displaying user interactions in real-time.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop a new project using the Micro:bit website.</li> <li>2. Construct a welcome message to display upon powering on the Microbit.</li> <li>3. Create a countdown sequence with visual cues using code.</li> <li>4. Implement a random delay function in the game for unpredictability.</li> <li>5. Create and utilise variables to store time stamps.</li> <li>6. Record and display player reaction time upon button press.</li> </ol>

## Week 4

### 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
<ol style="list-style-type: none"> <li>1. Develop skills in creating and manipulating variables within a Microbit project.</li> <li>2. Understand and apply the concept of random number generation in game development.</li> <li>3. Gain proficiency in programming button inputs to trigger specific actions.</li> <li>4. Learn to implement scoring systems and game over conditions in a game project.</li> <li>5. Enhance problem-solving skills by debugging and testing a game on a Microbit device.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create a new Microbit project and two variables for the game.</li> <li>2. Set up the start of the game with random numbers and display the number on the Microbit.</li> <li>3. Program the A button to guess if the next number is lower and score a point if the guess is correct.</li> <li>4. Reset the variables for the next round after a correct guess and display the new number on the Microbit.</li> <li>5. End the game if the guess is incorrect and display the final score.</li> <li>6. Program the B button to guess if the next number is higher, following the same rules as the A button.</li> <li>7. Download the game onto the Microbit and play it.</li> </ol>

## Week 5

### 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
<ol style="list-style-type: none"> <li>1. Develop skills in creating and managing a new Scratch project.</li> <li>2. Understand and apply the concept of adding and positioning sprites in Scratch.</li> <li>3. Gain proficiency in coding for sprite movement, interaction, and control using Scratch blocks.</li> <li>4. Learn to integrate and use a Microbit with Scratch for real-time control of sprites.</li> <li>5. Enhance critical thinking and problem-solving skills by identifying potential improvements to the game.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop a new Scratch project and remove the default cat sprite.</li> <li>2. Add and position the 'Paddle' sprite from the sprite library.</li> <li>3. Add the 'Soccer Ball' sprite from the sprite library.</li> <li>4. Set the X and Y coordinates to position the ball at the top center of the screen.</li> <li>5. Code the ball to move around the screen and bounce off the edges.</li> <li>6. Connect and configure a Microbit to the Scratch project.</li> <li>7. Code the paddle to move left and right by tilting the Microbit.</li> <li>8. Program the ball to bounce off the paddle when it touches it.</li> <li>9. Add the 'Stars' backdrop from the backdrop library.</li> <li>10. Draw a red line at the bottom of the screen for the game over line.</li> <li>11. Code the game to end when the ball touches the red game over line.</li> <li>12. Propose improvements to the game by adding to or changing the existing code.</li> </ol>

## Week 6

### Lesson: Microbit Compass and Thermometer

● Intermediate

🕒 60 mins

👥 Teacher/Student led

📝 Student Quiz

💡 Student Challenge

Prepare to guide students in creating a Microbit project that utilises the compass and temperature sensor. They will learn to create and set variables, program buttons, and use 'if then else' blocks. The lesson involves coding the Microbit to display cardinal directions based on its orientation and temperature readings. Students will also test their code using a simulator before sending it to their Microbit. Ensure familiarity with the makecode.com platform and basic coding concepts.

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
<ol style="list-style-type: none"> <li>1. Understand and utilise the compass and temperature sensor features of the Microbit.</li> <li>2. Develop proficiency in creating and setting variables in a Microbit project.</li> <li>3. Apply conditional logic to program Microbit buttons for specific functions.</li> <li>4. Test and debug code using the simulator before transferring to the Microbit.</li> <li>5. Interpret and display data from the Microbit's sensors in a user-friendly format.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop a new Microbit project using makecode.com.</li> <li>2. Create and set a 'direction' variable to store compass readings.</li> <li>3. Program the A button to display compass direction (N, S, E, W) based on 'direction' variable.</li> <li>4. Program the B button to display the current temperature reading.</li> <li>5. Test and debug the code using the simulator and then deploy it to the Microbit.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Develop a microbit voting system with separate voting and central microbits.</li> <li>2. Programme the voting microbits to cast a single 'Yes' or 'No' vote.</li> <li>3. Configure the central microbit to receive votes, count them, and reset the voting system.</li> <li>4. Implement a reset function on the voting microbits to allow for multiple rounds of voting.</li> <li>5. Enhance the voting system by adding a security feature to ensure the integrity of the votes.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop two separate projects on the MakeCode Microbit website for voting and central microbits.</li> <li>2. Programme the A and B buttons on the microbit to cast a single 'Yes' or 'No' vote.</li> <li>3. Set up the central microbit to receive votes, count them, and reset the voting system when needed.</li> <li>4. Configure the individual voting microbits to receive the 'Reset' signal from the central microbit.</li> <li>5. Display the vote results on the central microbit.</li> </ol>

## Week 8

### Lesson: Chase the Dot

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

In this lesson, students will create a game called 'Chase the Dot' using Microbit. They will learn to create a new project, define variables, create a function, and use gestures to control movements. The game involves two dots, a target and a chaser. The aim is for the chaser dot to catch the target dot, which moves to a random position each time it's caught. The lesson involves coding for dot creation, movement, scoring, and game initiation.

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
<ol style="list-style-type: none"> <li>1. Develop skills in creating and managing Microbit projects on makecode.com.</li> <li>2. Understand and apply the concept of variables in coding to create game sprites.</li> <li>3. Learn to create and use functions for repetitive tasks in a game scenario.</li> <li>4. Develop skills in using gestures to control game elements in a Microbit project.</li> <li>5. Understand and implement the concept of scoring and round systems in game development.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create and manipulate variables to store and control game sprites in a Microbit project.</li> <li>2. Develop a function to position a sprite at a random location on the edge of the screen.</li> <li>3. Implement a countdown timer and sound effects to enhance game play.</li> <li>4. Programme the Microbit to respond to tilt gestures to control sprite movement.</li> <li>5. Design a scoring system that responds to sprite interactions and triggers new rounds of play.</li> </ol>

## Week 9

### Lesson: Microbit Showcase

● Advanced

🕒 60 mins

👥 Teacher/Student led

Prepare students for a hands-on experience with Microbit, starting with project planning. Encourage them to brainstorm and sketch their ideas. Guide them through the coding process, reminding them to test their code frequently. Assist them in debugging, reinforcing that it's a normal part of coding. Finally, help them finalise their projects, ensuring their code is tidy and they can articulate what their project does. If possible, facilitate downloading the project onto a 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
<ol style="list-style-type: none"> <li>1. Develop a comprehensive plan for a Microbit project, including sketching ideas and outlining necessary steps.</li> <li>2. Apply coding skills to create and test a Microbit project, ensuring functionality and desired outcomes.</li> <li>3. Identify and rectify errors in the code through effective debugging techniques.</li> <li>4. Finalise the project by ensuring clean, organised code and providing a concise description of the project's purpose and functionality.</li> <li>5. Optional: Download and implement the project on a Microbit device, demonstrating understanding of the MakeCode editor.</li> </ol>	<ol style="list-style-type: none"> <li>1. Formulate a clear plan for a Microbit project, including a sketch and step-by-step instructions.</li> <li>2. Code a Microbit project using learned skills, with regular testing to ensure functionality.</li> <li>3. Debug the project code, identifying and rectifying errors to ensure the project works as intended.</li> <li>4. Finalise the project, ensuring the code is clean, organised, and accompanied by a concise description of the project and its functionality.</li> <li>5. Optional: Download the completed project onto a Microbit device using the MakeCode editor.</li> </ol>



# 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: <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> <li>• iPad/Tablet</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Master the use of MakeCode Arcade for game design and development.</li> <li>2. Develop skills in creating and controlling game sprites, including player and AI-controlled characters.</li> <li>3. Understand and implement game mechanics such as scoring, lives, collision effects, and game outcomes.</li> <li>4. Apply coding concepts to create interactive games with different themes and mechanics.</li> <li>5. Gain the ability to design, code, test, and refine a variety of games, culminating in a final game showcase.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create and control game sprites using MakeCode Arcade.</li> <li>2. Design and implement game mechanics such as movement, collision detection, scoring, and game over conditions.</li> <li>3. Understand and apply coding concepts to create interactive games, including sprite overlaps, game logic, and variable tracking.</li> <li>4. Develop a variety of games including arcade, platform, and battle arena games, demonstrating creativity and technical skills.</li> <li>5. Present a completed game project, demonstrating understanding of game design principles and coding concepts.</li> </ol>

# 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
<ol style="list-style-type: none"> <li>1. Understand and utilise MakeCode Arcade for creating games.</li> <li>2. Manipulate the Code Editor to build and modify game elements.</li> <li>3. Create and customise sprites for use in a game.</li> <li>4. Develop a tile map and implement walls for game navigation.</li> <li>5. Implement game mechanics such as projectiles, sprite movement, and life count.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the functions and features of MakeCode Arcade.</li> <li>2. Use the MakeCode Arcade code editor to create a new project and add a sprite.</li> <li>3. Manipulate the sprite's movements using the direction buttons in the simulator.</li> <li>4. Create and edit a tile map, including drawing walls and setting the camera to follow the sprite.</li> <li>5. Design and implement projectiles, including setting their direction and speed, and programming responses to overlaps with the player's sprite.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Develop skills in using MakeCode Arcade to create a game.</li> <li>2. Understand and apply the concept of sprites in game development.</li> <li>3. Implement controls for game characters using code.</li> <li>4. Apply the concept of randomisation in game elements for varied gameplay.</li> <li>5. Understand and implement game mechanics such as collision detection and life count.</li> </ol>	<ol style="list-style-type: none"> <li>1. Design and create a spaceship sprite using MakeCode Arcade.</li> <li>2. Control the spaceship's movement with arrow keys and set boundaries to prevent it from going off the screen.</li> <li>3. Set the number of lives for the spaceship.</li> <li>4. Create and design asteroid sprites that appear at random positions on the screen.</li> <li>5. Set the velocity of the asteroids to make them move across the screen.</li> <li>6. Implement a function to auto-destroy asteroids when they move off the screen.</li> <li>7. Program the game to detect when an asteroid hits the spaceship, causing it to lose a life and trigger a camera shake effect.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Develop skills in using MakeCode Arcade to create an interactive game.</li> <li>2. Understand how to create, control, and position player and enemy sprites.</li> <li>3. Learn to program game interactions such as shooting projectiles and detecting overlaps.</li> <li>4. Gain knowledge on how to keep score and end the game in MakeCode Arcade.</li> <li>5. Enhance problem-solving and debugging skills by experimenting with the code and adding new features.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create and control a player sprite in MakeCode Arcade.</li> <li>2. Generate enemy sprites at random positions.</li> <li>3. Program interactions between player and enemy sprites.</li> <li>4. Implement a scoring system for hitting targets.</li> <li>5. End the game when an enemy sprite hits the player sprite.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Develop understanding of MakeCode Arcade for game creation.</li> <li>2. Gain proficiency in creating and controlling game sprites.</li> <li>3. Learn to implement game mechanics such as scoring and lives.</li> <li>4. Understand how to detect and respond to sprite interactions.</li> <li>5. Apply coding skills to create a complete Space Shooter game.</li> </ol>	<ol style="list-style-type: none"> <li>1. Design and create a spaceship sprite in MakeCode Arcade.</li> <li>2. Control the spaceship sprite using arrow keys and prevent it from going off the screen.</li> <li>3. Set the number of lives for the spaceship.</li> <li>4. Create and program asteroids to fly in from the right side of the screen.</li> <li>5. Fire rockets from the spaceship when the A button is pressed.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Understand the basic concept and mechanics of platform games.</li> <li>2. Create and design a sprite character in a game environment.</li> <li>3. Implement movement controls for the sprite character.</li> <li>4. Apply the concept of gravity in a game setting.</li> <li>5. Design and create a game map with different elements such as ground, danger and goal tiles.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the concept of platform games and their mechanics.</li> <li>2. Create a new project on <a href="https://arcade.makecode.com">arcade.makecode.com</a> and design a sprite character.</li> <li>3. Implement sprite movement controls using code.</li> <li>4. Apply the concept of gravity to a sprite in a platform game.</li> <li>5. Design a game map with ground, danger, and goal tiles.</li> <li>6. Program a sprite to jump and move through the map.</li> <li>7. Implement game mechanics such as danger tiles and a goal tile.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Develop skills in creating an interactive game using MakeCode Arcade.</li> <li>2. Understand how to draw a map, create a character, and add movement and gravity effects.</li> <li>3. Learn to add obstacles and implement collision detection for game over scenarios.</li> <li>4. Gain knowledge on how to keep score and determine winning conditions in a game.</li> <li>5. Reflect on the game design process and the elements involved in creating an engaging game.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create a new arcade project on the MakeCode Arcade website.</li> <li>2. Draw a map for the Dino Jump game, including ground tiles, walls, and a finish tile.</li> <li>3. Create a dino sprite using the sprite editor and code.</li> <li>4. Implement gravity and movement for the dino sprite, making it fall to the ground and move forwards through the map.</li> <li>5. Program the A button to make the dino jump when it is touching the ground.</li> <li>6. Add trees to the map as obstacles for the dino to jump over.</li> <li>7. Implement game over functionality when the dino sprite hits a tree.</li> <li>8. Keep score based on how long the player can go without hitting a tree.</li> <li>9. Detect when the player reaches the end of the game and display a winning screen.</li> <li>10. Play and review the Dino Jump game, reflecting on the elements of game design learned during the lesson.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Develop a player-controlled sprite and an AI-controlled monster in a game using MakeCode Arcade.</li> <li>2. Create and manage a new project in MakeCode Arcade.</li> <li>3. Implement a health system for player and monster sprites.</li> <li>4. Develop a combat system where player and monster sprites can inflict damage on each other.</li> <li>5. Implement a win/lose condition based on the health of player and monster sprites.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create and control a player sprite using MakeCode Arcade, ensuring it moves smoothly within the screen boundaries.</li> <li>2. Program an AI-controlled monster sprite with randomized movement, simulating intelligent behavior.</li> <li>3. Implement a health system that tracks and displays the health values of both the player and the monster during the game.</li> <li>4. Develop a combat system that reduces player health upon collision with the monster and allows the player to shoot projectiles at the monster.</li> <li>5. Determine the game's winner by programming conditions that end the game when either the player's or monster's health reaches zero.</li> </ol>



## Week 8

### Lesson: Donut Rush

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

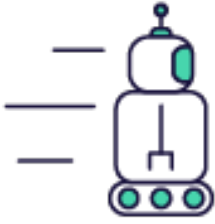
In this lesson, students will create an interactive game called 'Donut Rush' using MakeCode Arcade. They will learn to write code for creating game sprites, handling events like sprite overlaps, and controlling game logic. The lesson involves setting up the game, creating a new project, and defining 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
<ol style="list-style-type: none"> <li>1. Develop an understanding of game creation using MakeCode Arcade.</li> <li>2. Learn to create and manage variables in a gaming context.</li> <li>3. Understand the concept and application of functions in game development.</li> <li>4. Gain skills in handling events such as sprite overlaps and controlling game logic.</li> <li>5. Apply knowledge to create an interactive game with multiple levels and scoring system.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create a new project in MakeCode Arcade.</li> <li>2. Set up the game by creating a splash screen, setting up variables, and creating a player sprite.</li> <li>3. Create a function called 'startLevel' to organise the game's code.</li> <li>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.</li> <li>5. Create multiple donuts using a loop and place them randomly on the screen.</li> <li>6. Start the game by calling the 'startLevel' function.</li> <li>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.</li> <li>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.</li> <li>9. Wrap up the game and play it, aiming to collect as many donuts as possible within the time limit.</li> </ol>

# Module: Robotic Cars and Automation







This module delves into the fascinating world of robotics, starting with the basics of what a robot is, its history, and its future. It then transitions into practical, hands-on lessons where students build and program their own robotic cars using Microbits. Teachers should ensure students understand the theoretical aspects before moving onto the practical components. Encourage creativity and problem-solving as students navigate through building traffic lights, programming sensors, and even creating a robot car claw. The module culminates in a Robot Showcase, where students can display their creations.

Duration	Equipment
<p>10 weeks</p>	<p>Students can use any of these devices:</p> <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> </ul> <p>Required Equipment:</p> <ul style="list-style-type: none"> <li>• Microbit</li> <li>• Move Motor Car</li> <li>• Move Motor Klaw</li> <li>• Phillips Screwdriver</li> <li>• Traffic Lights Kit</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Understand the concept, history, and future of robotics and its impact on society.</li> <li>2. Develop practical skills in building and programming Microbit Traffic Lights.</li> <li>3. Acquire knowledge in constructing and programming a Move Motor Sensor Car.</li> <li>4. Learn to use sensors for line following, distance measurement, and object navigation in robotic cars.</li> <li>5. Gain proficiency in using a Microbit for remote control and communication between traffic lights and an autonomous car.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and explain the concept, history, and future of robotics.</li> <li>2. Construct and program traffic lights using a Microbit.</li> <li>3. Build and program a Move Motor Sensor Car to follow lines and navigate around objects.</li> <li>4. Utilise the accelerometer and radio in a Microbit to remotely control the Move Motor Car.</li> <li>5. Assemble, attach, and program the Move Motor Klaw to a Move Motor Car.</li> </ol>

# Week 1

## Lesson: What is a Robot?

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





In this lesson, we will learn about robots - what they are, what they look like, what they can do, and how they work. We'll explore different ways robots are used in different industries and environments. By the end of the lesson, we'll have a better understanding of what robotics is and how robots are changing the world around us.

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

- Chromebook/Laptop/PC

Learning Goals	Learning Outcomes
<ul style="list-style-type: none"> <li>• Understand the basic definition and purpose of a robot.</li> <li>• Identify different forms and appearances of robots and understand why they are designed in certain ways.</li> <li>• Recognize the various tasks that robots can perform and how they assist in different fields.</li> <li>• Comprehend the basic components of a robot and how they function together to perform tasks.</li> <li>• Identify different environments where robots are used and understand their role in these settings.</li> <li>• Appreciate the importance of robots in our society and how they contribute to efficiency, safety, and exploration.</li> </ul>	<ul style="list-style-type: none"> <li>• By the end of the lesson, students will be able to define what a robot is and describe its basic components such as sensors, motors, and controllers.</li> <li>• Students will be able to identify and describe different forms of robots, from humanoid to animal-inspired, and simple to complex designs.</li> <li>• Students will be able to explain the various tasks that robots can perform, from assisting humans in daily tasks to being used in scientific research or exploration.</li> <li>• Students will be able to explain how robots work, including how they use sensors to interact with their environment and make decisions.</li> <li>• Students will be able to identify and discuss the various environments where robots are used, including factories, hospitals, homes, and outer space.</li> <li>• Students will be able to articulate the importance of robots in society, including their role in increasing efficiency and productivity in various industries.</li> </ul>

## Lesson: History of robotics

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

In this lesson, we'll learn about the history of robotics, from the earliest mechanical devices to modern robots equipped with advanced sensors and artificial intelligence. We'll explore the impact of robotics on society and the future of this exciting field. By the end of the lesson, we'll have a better understanding of how robotics has changed the way we live and work.

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

- Chromebook/Laptop/PC

Learning Goals	Learning Outcomes
<ul style="list-style-type: none"> <li>• Understand the historical development of robotics from ancient times to the present day.</li> <li>• Identify key milestones and innovations in the history of robotics.</li> <li>• Recognize the impact of the Industrial Revolution on the development of robotics.</li> <li>• Appreciate the advancements in technology that have led to the modern robots we see today.</li> <li>• Analyze the impact of robotics on society, including the ethical considerations it raises.</li> <li>• Consider the potential future developments in robotics and their implications for society.</li> </ul>	<ul style="list-style-type: none"> <li>• Students will be able to trace the history of robotics from its earliest origins to the present day.</li> <li>• Students will be able to identify and describe the key developments in robotics during the Industrial Revolution.</li> <li>• Students will be able to explain the concept of a robot and identify the characteristics of early robots.</li> <li>• Students will be able to discuss the advancements in robotics technology, including the role of artificial intelligence and sensors.</li> <li>• Students will be able to analyze the impact of robotics on society, including its benefits, risks, and ethical implications.</li> <li>• Students will be able to predict future trends in robotics and discuss their potential societal implications.</li> </ul>

## Lesson: Future of robotics

● Beginner	🕒 20 mins	👥 Teacher/Student led	☰ Student Quiz
------------	-----------	-----------------------	----------------

In this lesson, we'll explore the exciting possibilities of the future of robotics. We'll learn about advancements in artificial intelligence, the promise of human-robot collaboration, and biologically inspired designs. We'll also consider the role of robots in space exploration and ethical considerations related to their use. By the end of the lesson, we'll have a better understanding of how robots will continue to transform the way we live and work in the future.





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

- Chromebook/Laptop/PC

Learning Goals	Learning Outcomes
<ul style="list-style-type: none"> <li>• Understand the role of artificial intelligence in the future of robotics and how it enables robots to learn, adapt, and make decisions.</li> <li>• Appreciate the concept of human-robot collaboration and how it can enhance our daily tasks and overall quality of life.</li> <li>• Recognize the principles of biologically inspired robotics and how nature influences the design and functionality of robots.</li> <li>• Understand the significance of robotics in space exploration and how it expands our understanding of the universe.</li> <li>• Identify the ethical considerations surrounding the use of robots and the need for responsible use and regulation.</li> <li>• Reflect on the potential impact of advancements in robotics on personal life and society as a whole.</li> </ul>	<ul style="list-style-type: none"> <li>• Upon completion of this lesson, students will be able to identify and explain the role of Artificial Intelligence in the future of robotics.</li> <li>• Students will be able to conceptualize and describe how human-robot collaboration can enhance daily tasks and improve safety.</li> <li>• Students will understand and discuss the concept of biologically inspired robotics and provide examples of its application.</li> <li>• Students will be able to explain the role of robots in space exploration and the advantages they offer over human explorers.</li> <li>• Students will be able to articulate ethical considerations related to the increased use of robots in society and propose potential solutions to these issues.</li> <li>• Upon reflection, students will be able to predict potential opportunities and challenges that the future of robotics may present to society.</li> </ul>

## Week 2

### 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
<ol style="list-style-type: none"> <li>1. Identify and gather necessary components for the Microbit Traffic Lights Kit.</li> <li>2. Understand and execute the process of unpacking and preparing the kit.</li> <li>3. Develop skills in assembling the stand for the traffic lights.</li> <li>4. Apply knowledge of Microbit to correctly align and attach it to the traffic lights.</li> <li>5. Demonstrate the ability to follow step-by-step instructions to complete a technical task.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and gather necessary components for the Microbit Traffic Lights Kit.</li> <li>2. Unpack and organise the Microbit Traffic Lights package contents.</li> <li>3. Assemble the stand from the provided parts in the kit.</li> <li>4. Align and attach the Microbit to the traffic lights using the correct hole configuration.</li> <li>5. Successfully complete the assembly of the Microbit Traffic Lights.</li> </ol>

### 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

<b>Learning Goals</b>	<b>Learning Outcomes</b>
<ol style="list-style-type: none"><li>1. Understand and apply the process of creating a new Microbit project.</li><li>2. Learn to add and utilise the Stopbit extension for programming traffic lights kit.</li><li>3. Gain skills in testing and troubleshooting the functionality of the lights.</li><li>4. Comprehend the concept of 'sequence' in coding and apply it to program traffic lights.</li><li>5. Develop proficiency in programming the sequence of traffic lights using on/off and state methods.</li></ol>	<ol style="list-style-type: none"><li>1. Create and manage a new Microbit project on makecode.com.</li><li>2. Add and utilise the "stopbit" extension to the Microbit project.</li><li>3. Test and troubleshoot the functionality of each light on the Microbit.</li><li>4. Understand and apply the concept of 'sequence' in coding to program traffic lights.</li><li>5. Program the sequence of traffic lights using on/off and state methods.</li></ol>

## Week 3

### 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
<ol style="list-style-type: none"> <li>1. Develop practical skills in assembling a Move Motor Sensor Car.</li> <li>2. Understand how to connect the Move Motor Sensor Car to Makecode.</li> <li>3. Acquire coding skills for controlling the motors, buzzer, and LEDs of the Move Motor Sensor Car.</li> <li>4. Learn to utilise the line following and distance sensors for navigation.</li> <li>5. Encourage exploration and creativity in coding for different movements and LED usage.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and organise components of the Move Motor Sensor Car kit.</li> <li>2. Assemble the Move Motor Sensor Car following the provided instructions.</li> <li>3. Connect the assembled car to Makecode and add the Move Motor Extension.</li> <li>4. Code the motors, buzzer, Zip LEDs, line following sensors, and distance sensor of the car.</li> <li>5. Apply learned skills to explore and create new movements and LED patterns.</li> </ol>

## Week 4

### 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
<ol style="list-style-type: none"> <li>1. Understand and apply the concept of programming a Microbit to control a Move Motor Car.</li> <li>2. Create and manipulate variables to store sensor values and control the car's movements.</li> <li>3. Implement conditional logic to guide the car's movements based on sensor readings.</li> <li>4. Use LEDs for visual feedback and enhance the functionality of the car.</li> <li>5. Experiment with code modifications to optimise the car's performance on different tracks.</li> </ol>	<ol style="list-style-type: none"> <li>1. Programme the Move Motor Car to follow a line track using a Microbit.</li> <li>2. Create a new project on the <a href="https://makecode.microbit.org">https://makecode.microbit.org</a> website.</li> <li>3. Add the kitronik-move-motor extension to the project and utilise the custom blocks to program the Move Motor car.</li> <li>4. Create and utilise variables to store values of the left and right line sensors and their difference.</li> <li>5. Set up the LEDs on the Move Motor car to light up different colours depending on the car's direction.</li> <li>6. Programme the car to turn right when the left sensor reads a higher darker value than the right sensor.</li> <li>7. Programme the car to turn left when the right sensor reads a higher darker value than the left sensor.</li> <li>8. Programme the car to move forwards when the left and right sensors have similar readings.</li> <li>9. Test the programmed car on a track and observe its autonomous driving.</li> <li>10. Tweak the code to improve the car's speed and performance on different tracks.</li> </ol>



## Week 5

### 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](https://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
<ol style="list-style-type: none"> <li>1. Understand the function and operation of an ultrasonic sensor.</li> <li>2. Develop proficiency in programming Microbits for specific tasks.</li> <li>3. Gain skills in using radio groups for communication between Microbits.</li> <li>4. Learn to measure distances using an ultrasonic sensor and a Microbit.</li> <li>5. Acquire the ability to display measurements on a separate Microbit.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and explain the function of an ultrasonic sensor and how it measures distance.</li> <li>2. Program the Microbit inside the Move Motor car to measure distances and send the measurements to another Microbit.</li> <li>3. Add necessary extensions to the project and set the units for measurement.</li> <li>4. Program the second Microbit to send a "measure" message and display the received measurement.</li> <li>5. Successfully test the functionality of the system by measuring and displaying distances.</li> </ol>

## Week 6

### 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
<ol style="list-style-type: none"> <li>1. Understand the function and operation of ultrasonic sensors.</li> <li>2. Develop skills in creating a new project using the kitronik-move-motor extension.</li> <li>3. Acquire the ability to program a sensor to measure distance.</li> <li>4. Learn to code a car to maintain a specific distance from an object.</li> <li>5. Enhance problem-solving skills by programming the car to reverse and maintain distance.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the function and operation of an ultrasonic sensor.</li> <li>2. Create a new project on the MakeCode Microbit website and add the necessary extension.</li> <li>3. Program the sensor to measure and display the distance to an object.</li> <li>4. Modify the code to make the car maintain a distance of 10cm from an object.</li> <li>5. Enhance the code to reverse the car until it is exactly 10cm away from an object.</li> <li>6. Program the car to free roam and avoid objects by stopping, reversing, and turning right when an object is detected within 10cm.</li> <li>7. Improve the code for better navigation and add lights for visual feedback.</li> </ol>

## Week 7

### 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
<ol style="list-style-type: none"> <li>1. Understand and apply the concept of radio communication between two Microbits.</li> <li>2. Programme a Microbit to send specific messages based on different gestures.</li> <li>3. Develop the ability to programme a Move Motor car to respond to different messages received.</li> <li>4. Test and debug the code to ensure the car responds correctly to the remote control.</li> <li>5. Extend the project by adding additional features such as LED light changes.</li> </ol>	<ol style="list-style-type: none"> <li>1. Programme a Microbit as a remote control to send directional commands.</li> <li>2. Programme a Microbit to receive and execute directional commands in a Move Motor car.</li> <li>3. Test and debug the code to ensure correct functioning of the remote-controlled car.</li> <li>4. Download and implement the code onto the Microbits.</li> <li>5. Extend the code to include LED light changes in response to different commands as an additional challenge.</li> </ol>

## Week 8

### 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
<ol style="list-style-type: none"> <li>1. Understand and apply the concept of radio communication between Microbits.</li> <li>2. Program a sequence of traffic light signals using code blocks.</li> <li>3. Develop skills to broadcast and receive specific messages based on traffic light states.</li> <li>4. Control the movement of a robot car based on received messages.</li> <li>5. Enhance problem-solving skills by modifying the code to respond based on the proximity of the car to the traffic lights.</li> </ol>	<ol style="list-style-type: none"> <li>1. Code a set of traffic lights to run through a sequence and broadcast the displayed light.</li> <li>2. Program a robot car to receive the broadcast and decide whether to stop or go based on the traffic light signal.</li> <li>3. Use the "stopbit" extension to create custom code blocks for programming the traffic lights kit.</li> <li>4. Program the car to move at different speeds or stop, depending on the received message from the traffic lights.</li> <li>5. Modify the code to make the car respond to the traffic lights based on its proximity to them.</li> </ol>

## Week 9

### Lesson: Attach the Move Motor Claw

 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 Claw 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 Claw

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop the ability to follow complex instructions with adult supervision.</li> <li>2. Understand the process of unpacking and organising components for assembly.</li> <li>3. Gain practical skills in using tools such as a small Phillips head screwdriver.</li> <li>4. Learn to assemble the Move Motor Claw and attach it to the Move Motor Car.</li> <li>5. Understand the flexibility of design in attaching the Claw either vertically or horizontally.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and gather necessary components for building the Move Motor Claw.</li> <li>2. Correctly open the package and organise its contents.</li> <li>3. Follow the provided instructions to assemble the Move Motor Claw.</li> <li>4. Successfully attach the Move Motor Claw to the Move Motor Car.</li> <li>5. Demonstrate safe and effective use of tools during assembly.</li> </ol>

### 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 Claw 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 Claw

**Learning Goals**

1. Understand the setup and requirements for a Move Motor Car with a Move Motor Klaw and a Microbit.
2. Develop skills in creating a new Microbit project using the provided website.
3. Gain knowledge about the kitronik-move-motor extension and how to add it to the project.
4. Comprehend the functioning of servos and their application in controlling the Move Motor Klaw.
5. Acquire the ability to program the claw to open and close using specific code blocks.

**Learning Outcomes**

1. Assemble a Move Motor Car with a Move Motor Klaw and a Microbit.
2. Create a new Microbit project on the specified website.
3. Add the kitronik-move-motor extension to the project.
4. Understand the function and operation of a servo in the Move Motor Klaw.
5. Program the claw to close and open using specific code blocks and test its functionality.
6. Modify the code to allow the claw to open and close gradually.
7. Explore other potential programming and usage possibilities for the claw.

# Module: Exploring Digital Art and Design







This module delves into the fascinating world of digital art and design. Teachers will guide students through the history and impact of digital art, the software and tools used in its creation, and the basics of software navigation. The module also covers the different types of brushes and tools, creating basic shapes, experimenting with brush strokes and effects, and an introduction to colour theory. The module concludes with a digital art showcase. Teachers should prepare by familiarising themselves with the module content and encouraging student participation and creativity throughout.

Duration	Equipment
7 weeks	Students can use any of these devices: <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> <li>• iPad/Tablet</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Develop an understanding of the history, forms, and impact of digital art.</li> <li>2. Gain familiarity with various digital art software and tools, and their unique functions.</li> <li>3. Master the basics of navigating and using the interface of digital art software.</li> <li>4. Understand the different types of brushes and tools used in digital art and their effective application.</li> <li>5. Apply the principles of color theory in the creation of digital art.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify the key characteristics and history of digital art, and discuss its impact on creativity.</li> <li>2. Understand and describe the functions of various digital art software and tools.</li> <li>3. Navigate and use the interface of digital art software efficiently, with a focus on Photopea.</li> <li>4. Differentiate between various types of brushes and tools in digital art, and use them effectively in creating artwork.</li> <li>5. Create basic shapes using digital art software and understand their role as the building blocks of artwork.</li> <li>6. Experiment with different brush strokes and effects to create unique digital art pieces.</li> <li>7. Understand the fundamental principles of colour theory and apply them effectively in digital art.</li> <li>8. Present a digital art piece, demonstrating the skills and knowledge acquired throughout the course.</li> </ol>

# Week 1

## Lesson: Introduction to Digital Art

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





Prepare to explain the concept of digital art, emphasising its basis in creativity and self-expression. Familiarise yourself with the history of digital art, from its inception in the 1950s to its current status. Highlight key developments and their impact on the art world. Discuss how digital art has expanded creative possibilities, enabling artists to experiment with new techniques and reach global audiences. Highlight the collaborative potential of digital art, particularly in virtual and augmented reality.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the concept and scope of digital art.</li> <li>2. Trace the historical development of digital art.</li> <li>3. Recognise the impact of digital art on creativity and artistic expression.</li> <li>4. Appreciate the role of technology in shaping and expanding the art world.</li> <li>5. Identify the potential of digital art in collaborative and global art projects.</li> </ol>	<ol style="list-style-type: none"> <li>1. Define digital art and its various forms.</li> <li>2. Trace the historical development of digital art from the 1950s to the present day.</li> <li>3. Identify the impact of technology on the evolution of digital art.</li> <li>4. Understand how digital art has expanded the avenues for creativity and expression.</li> <li>5. Recognise the role of digital art in global art sharing and collaboration.</li> </ol>

## Lesson: Overview of digital art software and tools

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

Prepare to introduce students to the world of digital art, its forms, and the software used to create it. Familiarise yourself with popular digital art software like Adobe Photoshop, Procreate, Photopea, and Corel Painter. Understand the difference between raster and vector software. Discuss the importance of digital art tools like drawing tablets, stylus pens, and computer mice. Explain the benefits of drawing tablets in digital art and the different types of stylus pen tips. Finally, discuss the various file formats for saving digital art.

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





- Chromebook/Laptop/PC
- iPad/Tablet



Learning Goals	Learning Outcomes
<ol style="list-style-type: none"><li>1. Understand the concept and types of digital art including digital paintings, illustrations, animations, and graphic design.</li><li>2. Identify and differentiate between popular digital art software such as Adobe Photoshop, Procreate, Photopea, and Corel Painter.</li><li>3. Understand the difference between raster and vector software and their respective applications.</li><li>4. Recognise the role and functionality of digital art tools including drawing tablets, stylus pens, and computer mice.</li><li>5. Understand the benefits of using drawing tablets and stylus pens, including their different tips, in creating digital art.</li><li>6. Identify common file formats for digital art and understand their specific uses and characteristics.</li></ol>	<ol style="list-style-type: none"><li>1. Identify and describe the key features of popular digital art software such as Adobe Photoshop, Procreate, Photopea, and Corel Painter.</li><li>2. Differentiate between raster and vector software, and identify their respective uses in digital art creation.</li><li>3. Recognise and explain the function of digital art tools including drawing tablets, stylus pens, and computer mice.</li><li>4. Understand the benefits and working of drawing tablets in digital art, including their pressure sensitivity and efficiency.</li><li>5. Identify different types of stylus pen tips and their uses in creating various effects on the digital canvas.</li><li>6. Understand and differentiate between common file formats for digital art, including JPEG, PNG, and PSD, and their respective uses.</li></ol>

## Week 2

### Lesson: Basic Navigation and Interface

 Beginner	 45 mins	 Teacher/Student led	 Student Quiz
---	---	---	--

This lesson will guide students through the basics of digital art software navigation and interface. Teachers should familiarise themselves with the Photopea software in advance. The lesson covers understanding the interface, creating a new project, and introduces navigation tools. Students will get hands-on experience using the paint, zoom, hand, and rotation tools, and will learn about the layers panel. The lesson concludes with an introduction to basic features such as brush settings, colour picker, layers, and selection tools, followed by practice exercises.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Grasp the structure and functionality of a digital art software interface.</li> <li>2. Initiate a new project using Photopea.</li> <li>3. Master the use of essential navigation tools within the software.</li> <li>4. Apply basic tools such as the Paint, Zoom, Hand, and Rotation tools effectively.</li> <li>5. Utilise the Layers Panel for efficient management and modification of artwork.</li> <li>6. Understand and apply basic features including brush settings, the colour picker, layers, and selection tools.</li> <li>7. Develop practical skills through exercises using these basic features.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and describe the layout and design of a digital art software interface.</li> <li>2. Create a new project using Photopea digital software.</li> <li>3. Utilise essential navigation tools to move around the canvas and access different features.</li> <li>4. Apply the Paint, Zoom, Hand, and Rotation tools in a digital art software.</li> <li>5. Manipulate the Layers Panel to work on different parts of artwork independently.</li> <li>6. Understand and use basic features such as brush settings, the colour picker, layers, and selection tools.</li> <li>7. Perform practice exercises to improve proficiency in using digital art software.</li> </ol>

## Week 3

### Lesson: Understanding the Different Types of Brushes and Tools

● Intermediate

🕒 45 mins

👥 Teacher/Student led

📝 Student Quiz

Prepare to guide students through the world of digital art, focusing on the different types of brushes and tools available in digital art software. Start with an introduction to digital art and its tools, then delve into the specifics of brushes and tools, explaining their unique features and uses. Encourage students to experiment with these tools, blending colours, creating different line widths and using negative space. Finally, task them with creating their own artwork using the discussed brushes and tools, encouraging peer feedback and appreciation.



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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Identify and describe the different types of brushes used in digital art.</li> <li>2. Identify and describe the different types of tools used in digital art.</li> <li>3. Understand the specific uses and effects of each brush and tool.</li> <li>4. Apply knowledge of brushes and tools in practical exercises.</li> <li>5. Create original artwork using a variety of brushes and tools.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and differentiate between various types of brushes used in digital art software.</li> <li>2. Recognise and distinguish between different tools available in digital art software.</li> <li>3. Understand and apply the specific uses of each brush and tool in digital art creation.</li> <li>4. Experiment with different combinations of brushes and tools to create unique effects in artwork.</li> <li>5. Create a piece of digital artwork utilising the learned brushes and tools effectively.</li> </ol>

## Week 4

### Lesson: Creating basic shapes

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

For this lesson, teachers should familiarise themselves with the basic shape tools in Photopea, including the rectangle, ellipse, and polygon tools. They should also understand how to create perfect circles, squares, rectangles, and triangles using these tools. Additional shape tools and their uses should be explored. Teachers should prepare some fun projects for students to practice their skills, such as creating digital art pieces, designing website layouts, or creating infographics.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the role and importance of shapes in digital art.</li> <li>2. Identify and utilise basic shape tools in Photopea.</li> <li>3. Create perfect circles, squares, rectangles, and triangles using Photopea.</li> <li>4. Explore additional shape tools and their applications in Photopea.</li> <li>5. Apply learned skills to create unique designs and graphics in fun projects.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and utilise various basic shape tools in Photopea.</li> <li>2. Create a perfect circle using the Ellipse Tool in Photopea.</li> <li>3. Formulate a square or rectangle using the Rectangle Tool in Photopea.</li> <li>4. Construct a triangle using the Polygon Tool in Photopea.</li> <li>5. Apply additional shape tools and features in Photopea to create complex shapes and designs.</li> </ol>

## Week 5

### Lesson: Experimenting with Different Brush Strokes and Effects

● Intermediate

🕒 45 mins

👥 Teacher/Student led

📝 Student Quiz

For this lesson, ensure familiarity with different types of digital art brushes and their effects. Prepare to demonstrate various brush strokes and techniques, adjusting brush settings for unique effects. Encourage students to experiment with these techniques in creating their own digital art piece. Be ready to guide them through saving their artwork correctly. Finally, emphasise the importance of regular practice, taking on challenges, seeking feedback, and learning from others to improve their digital art skills.



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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the characteristics and capabilities of different brush types and effects in digital art.</li> <li>2. Master various brush strokes and techniques to create diverse effects and textures.</li> <li>3. Apply basic and advanced brush settings to manipulate the behaviour of brushes and create complex effects.</li> <li>4. Create a unique digital art piece using a variety of brushes, strokes, and techniques.</li> <li>5. Develop strategies for continuous practice and improvement in digital art creation.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and describe different types of digital art brushes and their effects.</li> <li>2. Apply various brush strokes and techniques to create different effects in digital art.</li> <li>3. Utilise basic and advanced brush settings to manipulate the behaviour of digital brushes.</li> <li>4. Create a digital art piece using a variety of brushes, strokes, and effects.</li> <li>5. Save digital artwork effectively and practice techniques for continuous improvement.</li> </ol>

## Week 6

### Lesson: Introduction to Color Theory

 Intermediate	 40 mins	 Teacher/Student led	 Student Quiz
---	---	---	--

Prepare to introduce students to the fundamentals of colour theory, starting with the colour wheel and primary, secondary, and tertiary colours. Discuss complementary and analogous colour schemes, and the concepts of hue, saturation, and brightness. Demonstrate how to adjust these properties in Photopea. Explain the role of contrast in digital art, and how to achieve high and low contrast. Finally, guide students in applying colour theory to their own digital art.




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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the basic principles of color theory, including the color wheel and the primary, secondary, and tertiary colors.</li> <li>2. Recognise and apply complementary and analogous color schemes in artwork.</li> <li>3. Comprehend and manipulate hue, saturation, and brightness in digital art.</li> <li>4. Apply high and low contrast techniques to create visual interest in digital art.</li> <li>5. Integrate color theory principles into digital art to create harmonious and visually appealing designs.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify primary, secondary, and tertiary colours on the colour wheel.</li> <li>2. Distinguish between complementary and analogous colour schemes.</li> <li>3. Define and differentiate hue, saturation, and brightness in digital art.</li> <li>4. Adjust hue, saturation, and brightness in Photopea.</li> <li>5. Apply high and low contrast techniques in digital art.</li> </ol>

## Week 7

### Lesson: Using Color in Digital Art

 Intermediate	 45 mins	 Teacher/Student led	 Student Quiz
---	---	---	--

Prepare to guide students through understanding various colour schemes and their impact on digital art. Ensure familiarity with monochromatic, analogous, complementary, and triadic schemes. Discuss how colour influences mood and emotions in art. Demonstrate application of colour in digital art using software like Adobe Photoshop or Procreate. Discuss techniques for effective colour application and tips for choosing the right colour palette. Encourage students to consider the context and purpose of their artwork when selecting colours.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand and apply different colour schemes in digital art: monochromatic, analogous, complementary, and triadic.</li> <li>2. Use monochromatic colour schemes to create depth and dimension in artwork.</li> <li>3. Apply analogous colour schemes to create a natural and cohesive look in artwork.</li> <li>4. Utilise complementary colour schemes to create high contrast and dynamic effects in artwork.</li> <li>5. Implement triadic colour schemes to create vibrant and balanced effects in artwork.</li> <li>6. Recognise the impact of colour on the mood and emotions of an artwork and use it effectively.</li> <li>7. Apply colour in digital art using software tools and techniques.</li> <li>8. Use different techniques for applying colour effectively in digital art, including contrast, depth, and directing viewer's attention.</li> <li>9. Select the right colour palette for artwork considering the context, purpose, and desired message or story.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and describe the four main types of colour schemes used in art and design.</li> <li>2. Apply a monochromatic colour scheme in digital art to create a calm and peaceful atmosphere.</li> <li>3. Use an analogous colour scheme in digital art to create a natural and cohesive look.</li> <li>4. Implement a complementary colour scheme in digital art for a high contrast and dynamic effect.</li> <li>5. Apply a triadic colour scheme in digital art for a vibrant and balanced effect.</li> <li>6. Use colour to influence and convey mood and emotions in digital art.</li> <li>7. Apply colour in digital art using software tools and adjust colour balance, hue, saturation, and brightness to achieve the desired effect.</li> <li>8. Use different techniques for applying colour effectively in digital art, including using light and dark values for contrast and depth.</li> <li>9. Select the right colour palette for digital art based on the context and purpose of the artwork.</li> </ol>

## Week 8

### Lesson: Digital Art Showcase

● Advanced

🕒 60 mins

Prepare for a digital art showcase, guiding students through brainstorming, tool selection, creation, refinement, and presentation of their artwork. Encourage creativity, patience, and regular saving of work. Promote critical evaluation and adjustments, fostering a willingness to experiment. Finally, ensure students feel proud to share their creations, either in class or on social media.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop a concept and plan for a digital art piece.</li> <li>2. Select appropriate digital art software and tools to create desired effects.</li> <li>3. Create a digital art piece, demonstrating patience and attention to detail.</li> <li>4. Review and refine the digital artwork, demonstrating a willingness to experiment and improve.</li> <li>5. Present the final artwork in a public setting, demonstrating confidence and pride in personal creativity.</li> </ol>	<ol style="list-style-type: none"> <li>1. Generate and articulate creative ideas for a digital art piece.</li> <li>2. Select appropriate digital art software and tools to achieve desired effects.</li> <li>3. Create a digital artwork, demonstrating patience and regular saving habits.</li> <li>4. Review and refine digital artwork, showing willingness to experiment and adjust.</li> <li>5. Present final digital artwork to peers or on social media platforms.</li> </ol>



# Module: Web Design Basics







This module introduces students to the basics of web design, starting with HTML and progressing to CSS. Teachers should ensure students understand the structure of a web page and the role of HTML elements. Encourage hands-on practice and experimentation with code. As the module progresses, students will learn to create complex tables, forms, and embed multimedia elements. The final modules introduce CSS, covering text and font styling, the box model, and website layout. Teachers should reinforce learning with practical exercises and real-world examples.

Duration	Equipment
9 weeks	Students can use any of these devices: <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> <li>• iPad/Tablet</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Understand and apply basic HTML elements to create structured web pages.</li> <li>2. Design and implement complex HTML tables and lists.</li> <li>3. Create interactive forms using basic and advanced HTML input types.</li> <li>4. Embed multimedia elements into web pages using HTML5.</li> <li>5. Utilise CSS for styling web pages, manipulating text and fonts, and creating website layouts.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and apply basic HTML elements such as headings, paragraphs, breaks, images, and links to structure a webpage.</li> <li>2. Create and manipulate complex HTML tables using advanced features like rowspan and colspan.</li> <li>3. Design and code interactive forms using HTML elements like &lt;input&gt;, &lt;label&gt;, and &lt;button&gt; and apply advanced input types.</li> <li>4. Embed audio and video files into web pages using HTML5 multimedia elements.</li> <li>5. Utilise CSS to style web pages, including text and fonts, and create a basic website layout.</li> </ol>

# 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
<ol style="list-style-type: none"> <li>1. Understand the purpose and structure of HTML in web development.</li> <li>2. Identify and describe the key components of a web page structure.</li> <li>3. Recognise and use basic HTML elements in coding.</li> <li>4. Explain the role of web browsers in interpreting and displaying HTML code.</li> <li>5. Apply knowledge to write and modify basic HTML code to create a simple web page.</li> </ol>	<ol style="list-style-type: none"> <li>1. Define HTML and its role in web page creation.</li> <li>2. Identify and explain the structure of a web page using HTML.</li> <li>3. Recognise and describe HTML elements and their functions.</li> <li>4. Understand how web browsers interpret and display HTML code.</li> <li>5. Write and run basic HTML code to create a simple web page.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Understand and apply HTML heading tags from h1 to h6.</li> <li>2. Create and format paragraphs using the p tag.</li> <li>3. Implement line breaks in HTML text using the br tag.</li> <li>4. Insert and manipulate images using the img tag and its attributes.</li> <li>5. Create hyperlinks to other web pages using the a tag and its attributes.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and use HTML heading tags from &lt;h1&gt; to &lt;h6&gt;.</li> <li>2. Define and implement paragraphs using the &lt;p&gt; tag.</li> <li>3. Insert line breaks in HTML documents using the &lt;br&gt; tag.</li> <li>4. Embed images in HTML using the &lt;img&gt; tag and its attributes.</li> <li>5. Create hyperlinks using the &lt;a&gt; tag and understand the use of its attributes.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Understand the purpose and structure of HTML tables.</li> <li>2. Identify and use HTML tags to create table headers, bodies, and footers.</li> <li>3. Create table rows and columns using appropriate HTML tags.</li> <li>4. Apply HTML coding to create a personalised table with multiple rows and columns.</li> <li>5. Develop skills to troubleshoot and correct HTML table coding errors.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and explain the structure and purpose of HTML tables.</li> <li>2. Create a basic HTML table using the &lt;table&gt;, &lt;tr&gt;, and &lt;td&gt; tags.</li> <li>3. Use &lt;thead&gt;, &lt;tbody&gt;, and &lt;tfoot&gt; tags to define the header, body, and footer of a table.</li> <li>4. Develop a multi-row HTML table with specific content in each cell.</li> <li>5. Apply CSS styling to HTML tables and their cells.</li> </ol>

### 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
<ol style="list-style-type: none"> <li>1. Master the creation of basic HTML tables with multiple rows and columns.</li> <li>2. Understand and apply the concept of table headers for better data context.</li> <li>3. Learn to use the 'rowspan' attribute to merge cells vertically.</li> <li>4. Learn to use the 'colspan' attribute to merge cells horizontally.</li> <li>5. Develop the ability to add footer rows to tables for summary information.</li> </ol>	<ol style="list-style-type: none"> <li>1. Construct a basic HTML table with multiple rows and columns.</li> <li>2. Integrate a header row into the table for column labelling.</li> <li>3. Apply the 'rowspan' attribute to merge cells vertically.</li> <li>4. Utilise the 'colspan' attribute to merge cells horizontally.</li> <li>5. Add a footer row to the table for summary information.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Understand the purpose and usage of HTML lists.</li> <li>2. Code ordered and unordered lists in HTML.</li> <li>3. Manipulate the numbering of ordered lists and bullet points of unordered lists.</li> <li>4. Create nested lists in HTML.</li> <li>5. Apply learned skills to code a personal list.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and differentiate between ordered and unordered HTML lists.</li> <li>2. Code ordered lists using the <code>&lt;ol&gt;</code> and <code>&lt;li&gt;</code> tags.</li> <li>3. Manipulate the numbering type of ordered lists using the 'type' attribute.</li> <li>4. Code unordered lists using the <code>&lt;ul&gt;</code> and <code>&lt;li&gt;</code> tags.</li> <li>5. Alter the bullet point type of unordered lists using the 'type' attribute.</li> <li>6. Create nested lists within both ordered and unordered lists.</li> <li>7. Apply learned skills to code a personalised list.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Understand the purpose and structure of HTML forms</li> <li>2. Create a form container using the <code>&lt;form&gt;</code> element</li> <li>3. Add and configure input fields for user data collection</li> <li>4. Implement labels for enhanced accessibility and usability</li> <li>5. Include a textarea for multi-line text input</li> <li>6. Add a submit button to enable form submission</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and apply the HTML <code>&lt;form&gt;</code> element to create a form container.</li> <li>2. Create and utilise <code>&lt;input&gt;</code> fields for text and email data collection.</li> <li>3. Implement <code>&lt;label&gt;</code> elements for improved form accessibility and usability.</li> <li>4. Add a <code>&lt;textarea&gt;</code> element for multi-line text input.</li> <li>5. Include a <code>&lt;button&gt;</code> with a <code>type</code> attribute set to submit to finalise form creation.</li> </ol>

### 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
<ol style="list-style-type: none"><li>1. Understand and utilise advanced HTML input types including number, date, and colour.</li><li>2. Create a number input field for capturing numerical data such as age.</li><li>3. Implement a date input field for selecting specific dates.</li><li>4. Use a colour input field for selecting a colour from a colour picker.</li><li>5. Enhance a form by integrating advanced input types to improve functionality and user experience.</li></ol>	<ol style="list-style-type: none"><li>1. Understand and apply the number input type in HTML forms.</li><li>2. Understand and apply the date input type in HTML forms.</li><li>3. Understand and apply the color input type in HTML forms.</li><li>4. Create HTML forms utilising advanced input types.</li><li>5. Enhance user experience by implementing advanced input types in forms.</li></ol>

## 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
<ol style="list-style-type: none"> <li>1. Identify and understand the different audio and video formats supported by HTML5.</li> <li>2. Embed an audio file into a web page using the HTML5 <code>&lt;audio&gt;</code> tag.</li> <li>3. Embed a video file into a web page using the HTML5 <code>&lt;video&gt;</code> tag.</li> <li>4. Preview and test the functionality of embedded audio and video players.</li> <li>5. Reflect on the process and importance of embedding multimedia elements in web pages.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and understand the common audio and video formats supported by HTML5 and their compatibility with various web browsers.</li> <li>2. Embed an audio file into a web page using the HTML5 <code>&lt;audio&gt;</code> tag and provide an MP3 file as the source.</li> <li>3. Embed a video file into a web page using the HTML5 <code>&lt;video&gt;</code> tag and provide an MP4 file as the source.</li> <li>4. Preview and test the functionality of the embedded audio and video players, and experiment with different video dimensions.</li> <li>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.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Understand the purpose and function of CSS in web development.</li> <li>2. Identify and apply CSS rules including selectors and declaration blocks.</li> <li>3. Differentiate between element, ID, and class selectors in CSS.</li> <li>4. Apply CSS styles directly to HTML elements using the style property.</li> <li>5. Practice creating and applying CSS classes to HTML elements.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the purpose and function of CSS in web development.</li> <li>2. Identify and apply CSS rules including selectors and declaration blocks.</li> <li>3. Use CSS to style HTML elements using element, ID, and class selectors.</li> <li>4. Apply CSS properties directly to HTML elements using the style property.</li> <li>5. Practice writing CSS code through exercises and understand how it affects the appearance of HTML elements.</li> </ol>

### 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
<ol style="list-style-type: none"> <li>1. Understand the CSS Box Model and its components: margins, borders, padding, and content.</li> <li>2. Apply different styles, widths, and colours to CSS borders.</li> <li>3. Manipulate padding in CSS to create space around content within the border.</li> <li>4. Use CSS margins to create space around elements, outside the border.</li> <li>5. Perform exercises to apply CSS Box Model properties to HTML elements.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and explain the components of the CSS Box Model.</li> <li>2. Apply CSS properties to create and modify borders on HTML elements.</li> <li>3. Use CSS properties to set border styles, widths, and colours.</li> <li>4. Apply CSS padding to create space around content within an element.</li> <li>5. Use CSS margins to create space around HTML elements.</li> </ol>



## 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
<ol style="list-style-type: none"> <li>1. Understand and apply CSS properties to style text colour and background colour.</li> <li>2. Manipulate text alignment using CSS.</li> <li>3. Use CSS to add or remove text decorations.</li> <li>4. Transform text to uppercase, lowercase or capitalised format using CSS.</li> <li>5. Apply CSS properties to adjust text spacing and add text shadow.</li> </ol>	<ol style="list-style-type: none"> <li>1. Apply CSS properties to style text colour and background colour.</li> <li>2. Align text using CSS properties.</li> <li>3. Decorate text using underline, overline and line-through CSS properties.</li> <li>4. Transform text to uppercase, lowercase and capitalise using CSS properties.</li> <li>5. Apply CSS properties to set text spacing and add text shadow.</li> </ol>

### 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
<ol style="list-style-type: none"> <li>1. Understand the importance of font selection in CSS and how it impacts user experience.</li> <li>2. Apply the 'font-family' property to set specific fonts for text elements.</li> <li>3. Use 'font-family' to specify fallback fonts when the primary font is unavailable.</li> <li>4. Manipulate 'font-weight', 'font-size', and 'font-style' properties to modify the appearance of text.</li> <li>5. Implement learned CSS font properties in a practical exercise.</li> </ol>	<ol style="list-style-type: none"> <li>1. Apply the CSS property 'font-family' to set specific fonts for text.</li> <li>2. Specify 'fallback' fonts using the 'font-family' CSS property.</li> <li>3. Manipulate the weight of the font using the 'font-weight' CSS property.</li> <li>4. Adjust the size of the font using the 'font-size' CSS property.</li> <li>5. Change the style of the font using the 'font-style' CSS property.</li> </ol>

## 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
<ol style="list-style-type: none"> <li>1. Understand the structure of a website layout including header, content, and footer.</li> <li>2. Develop skills to code and style a website header using HTML and CSS.</li> <li>3. Learn to create different content layouts such as one-column, two-column, and three-column layouts.</li> <li>4. Gain proficiency in using CSS to set column widths and layout.</li> <li>5. Acquire knowledge to code and style a website footer using HTML and CSS.</li> </ol>	<ol style="list-style-type: none"> <li>1. Identify and code common website layout areas: header, content, and footer using CSS.</li> <li>2. Construct and style a website header with logo and navigation menu.</li> <li>3. Design and implement one, two, and three column content layouts.</li> <li>4. Manipulate column widths and padding to achieve desired layout.</li> <li>5. Create and style a website footer with company information and secondary links.</li> <li>6. Combine all elements to create a cohesive website layout.</li> </ol>

# Module: Dynamic Web Design







This module covers the fundamentals of dynamic web design, starting with an overview of HTML, CSS, and JavaScript. It then progresses to setting up essential tools, scripting and DOM manipulation, dynamic form validation, and integrating external libraries and APIs. The module concludes with students creating an interactive quiz game, a weather web app, and a web showcase. Teachers should encourage active learning through hands-on coding exercises and challenges. Familiarity with CodePen, jQuery, and APIs is beneficial.

Duration	Equipment
8 weeks	Students can use any of these devices: <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> <li>• iPad/Tablet</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Understand and apply the interaction of HTML, CSS, and JavaScript in creating dynamic web pages.</li> <li>2. Set up and utilise essential web development tools including code editors, browser developer tools, and debugging consoles.</li> <li>3. Master advanced scripting techniques for DOM manipulation, including creating, deleting, and modifying HTML elements.</li> <li>4. Implement dynamic form validation with custom messages using JavaScript, providing real-time feedback for various input types.</li> <li>5. Integrate external libraries such as jQuery and APIs to pull dynamic data into web pages.</li> <li>6. Design and develop interactive web applications, such as a quiz game and a weather web app, incorporating real-time data and user interaction.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and apply the interaction of HTML, CSS, and JavaScript to create dynamic web pages.</li> <li>2. Set up and utilise essential web development tools including a code editor, browser developer tools, and the console for debugging.</li> <li>3. Manipulate the DOM using advanced scripting techniques such as creating, deleting, or modifying HTML elements based on certain conditions or inputs.</li> <li>4. Implement dynamic form validation with custom validation messages using JavaScript, providing real-time feedback as users fill out forms and validating different input types.</li> <li>5. Integrate external libraries like jQuery and APIs to pull dynamic data into web pages.</li> <li>6. Develop an interactive quiz game that checks answers, provides feedback, and incorporates timers, score trackers, and dynamic question loading.</li> <li>7. Create a Weather Web App that pulls real-time weather data based on a location, and displays it in an engaging and interactive manner.</li> <li>8. Present a web showcase demonstrating the skills and knowledge acquired throughout the course.</li> </ol>

# Week 1

## Lesson: Overview of how HTML, CSS, and JavaScript Interact

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


Prepare to explain the analogy of web development to building a house, with HTML, CSS, and JavaScript as the structure, design, and functionality respectively. Ensure understanding of the basic structures of HTML, CSS, and JavaScript, including their syntax and usage. Highlight the synergy of these three languages in creating dynamic web pages. Be ready to discuss real-life applications, such as creating a web-based quiz, to illustrate their interactivity. Conclude by emphasising the importance of proficiency in all three components for effective web development.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand the roles of HTML, CSS, and JavaScript in web development.</li> <li>2. Comprehend the basic structure and syntax of HTML, CSS, and JavaScript.</li> <li>3. Apply CSS styles in different ways: inline, internal, and external.</li> <li>4. Recognise common uses of JavaScript in enhancing web interactivity.</li> <li>5. Appreciate the synergy of HTML, CSS, and JavaScript in creating dynamic web pages.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand the roles of HTML, CSS, and JavaScript in web development.</li> <li>2. Identify the basic structure and elements of an HTML document.</li> <li>3. Apply CSS to control the appearance of a webpage.</li> <li>4. Use JavaScript to add interactivity to web pages.</li> <li>5. Integrate HTML, CSS, and JavaScript to create dynamic web pages.</li> </ol>

## Lesson: Setting up Essential Tools

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

This lesson guides students through setting up essential web development tools. They'll learn about code editors, browser developer tools, and the console for debugging. Students will explore CodePen, an online code editor, and create a basic webpage using HTML, CSS, and JavaScript. They'll also add a button with an onclick event. The lesson concludes with a wrap-up and encouragement for further practice.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"><li>1. Gain proficiency in using web development environments and essential tools such as code editors, browser developer tools, and the console for debugging.</li><li>2. Understand the features and benefits of using CodePen as an online code editor.</li><li>3. Develop skills to inspect, debug, and optimise code using browser developer tools.</li><li>4. Learn to use the console for identifying and resolving issues in JavaScript code.</li><li>5. Apply knowledge to create a basic webpage on CodePen, incorporating HTML, CSS, and JavaScript, and adding interactive elements like buttons with onclick events.</li></ol>	<ol style="list-style-type: none"><li>1. Identify and utilise essential web development tools including code editors, browser developer tools, and the console for debugging.</li><li>2. Select and use CodePen as an online code editor for writing and previewing HTML, CSS, and JavaScript in real-time.</li><li>3. Inspect and debug code using browser developer tools and the console.</li><li>4. Create and edit a basic webpage on CodePen using HTML, CSS, and JavaScript.</li><li>5. Add interactive elements to a webpage, such as a button with an onclick event, using JavaScript.</li></ol>

## Week 2

### Lesson: Scripting and DOM Manipulation

● Intermediate

🕒 60 mins

👥 Teacher/Student led

📝 Student Quiz

💡 Student Challenge

Prepare to guide students through a hands-on exploration of scripting and DOM manipulation. They'll set up a CodePen project, create HTML structures, and add JavaScript functions. They'll learn to use 'onclick' attributes, event listeners, and manipulate text colour and size. The lesson concludes with challenges to create small text and remove elements, reinforcing their understanding of DOM manipulation.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand and apply the concept of DOM manipulation using JavaScript.</li> <li>2. Develop skills in creating and modifying HTML elements dynamically.</li> <li>3. Gain proficiency in handling events using JavaScript, including click and mouseover events.</li> <li>4. Learn to use JavaScript to alter CSS properties of HTML elements.</li> <li>5. Apply problem-solving skills to complete coding challenges related to DOM manipulation.</li> </ol>	<ol style="list-style-type: none"> <li>1. Set up and utilise CodePen for HTML and JavaScript scripting.</li> <li>2. Create and manipulate HTML structure using JavaScript.</li> <li>3. Implement JavaScript functions to dynamically add elements to a webpage.</li> <li>4. Utilise event listeners to trigger JavaScript functions.</li> <li>5. Manipulate CSS properties of HTML elements through JavaScript.</li> </ol>

## Week 3

### Lesson: Dynamic Form Validation with JavaScript

● Advanced

🕒 60 mins

👥 Teacher/Student led

☰ Student Quiz

💡 Student Challenge

Prepare for a hands-on lesson on dynamic form validation using JavaScript. Familiarise yourself with the CodePen environment, as students will be setting up their projects there. The lesson will guide students through creating a form, styling it with CSS, and adding JavaScript for validation. They will learn to validate fields for name, email, and password, ensuring the correct length and format. The lesson concludes with a challenge to add an age field and validate it.






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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Understand how to set up a project on CodePen for HTML, CSS, and JavaScript development.</li> <li>2. Create and style a form using HTML and CSS.</li> <li>3. Implement JavaScript code to validate form fields for specific requirements.</li> <li>4. Test form validation and handle form submission using JavaScript.</li> <li>5. Extend JavaScript validation to new form fields, demonstrating adaptability of skills.</li> </ol>	<ol style="list-style-type: none"> <li>1. Set up a project environment in CodePen.</li> <li>2. Create a form with name, email, and password fields using HTML.</li> <li>3. Style the form using CSS for better visual appeal.</li> <li>4. Implement JavaScript code to prevent form submission and enable validation.</li> <li>5. Validate the name field to ensure it is at least 3 characters long.</li> <li>6. Validate the email field to ensure it contains '@' and '.' characters.</li> <li>7. Validate the password field to ensure it is at least 8 characters long and contains at least one number and one letter.</li> <li>8. Test the form by entering different values and checking if the validation works as expected.</li> <li>9. Add an age field and validate it to ensure the person is over 13.</li> </ol>

## Week 4

### Lesson: Integrating External Libraries and APIs

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

Prepare to guide students through the process of integrating external libraries and APIs into a project. The lesson involves setting up a project on CodePen, adding jQuery, creating an HTML structure, understanding jQuery syntax, selectors, and events, adding a click event listener, fetching weather data with an API, displaying the weather data, and extending the functionality of the weather app. The lesson concludes with wrapping up the weather app and encouraging students to explore more advanced features and APIs.

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






- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop proficiency in setting up a new project on CodePen.</li> <li>2. Gain understanding and practical skills in integrating jQuery into a project.</li> <li>3. Master the creation of HTML structures and the application of jQuery syntax and selectors.</li> <li>4. Learn to handle jQuery events and implement event listeners.</li> <li>5. Acquire skills in fetching data from external APIs and integrating it into a web application.</li> </ol>	<ol style="list-style-type: none"> <li>1. Set up a new project using CodePen and integrate jQuery library.</li> <li>2. Create a basic HTML structure for a web application.</li> <li>3. Understand and apply jQuery syntax to select and manipulate HTML elements.</li> <li>4. Handle user interactions using jQuery event listeners.</li> <li>5. Fetch and display real-time weather data from an external API.</li> </ol>



## Week 5

### Lesson: Interactive Quiz Game

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

This lesson guides students through creating an interactive quiz game using HTML, CSS, and JavaScript. They'll learn how to set up a project on CodePen, structure HTML for the game, style it with CSS, and add functionality with JavaScript. The lesson includes adding a jQuery library, setting up questions, variables, and functions to display questions, check answers, and display scores. It concludes with challenges to add a timer and different difficulty levels to the game.




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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop skills in setting up a coding project using CodePen.</li> <li>2. Understand and apply HTML structure to create an interactive quiz game.</li> <li>3. Apply CSS styling to enhance the visual presentation of the quiz game.</li> <li>4. Utilise jQuery library to manipulate HTML elements and handle user interactions.</li> <li>5. Create and manipulate JavaScript arrays and objects to store quiz questions and answers.</li> </ol>	<ol style="list-style-type: none"> <li>1. Create a new project on CodePen and add HTML structure for an interactive quiz game.</li> <li>2. Apply CSS styling to HTML elements for visual enhancement.</li> <li>3. Integrate the jQuery library into the project for dynamic features.</li> <li>4. Set up an array of question objects and variables for tracking quiz progress.</li> <li>5. Display questions and answer options dynamically, and check user's answers for correctness.</li> <li>6. Implement functionality to move to the next question after an answer is selected.</li> <li>7. Display the user's score after all questions have been answered.</li> <li>8. Enhance the quiz game with a timer for each question.</li> <li>9. Add different difficulty levels to the quiz game for varied user experience.</li> </ol>

## Week 6

### Lesson: Weather Web App

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

In this lesson, students will create a Weather Web App using HTML, CSS, and JavaScript. They will learn how to fetch and display real-time weather data using APIs and jQuery. The lesson will guide them through setting up the project, adding jQuery and Fontawesome, creating the HTML structure, styling the structure and headings, initializing JavaScript, fetching and displaying weather data, adding a unit toggle, and testing the app. They will also be encouraged to enhance their app by adding additional features such as a search bar and displaying more weather information.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Develop a functional weather web application using HTML, CSS, and JavaScript.</li> <li>2. Utilise jQuery for efficient manipulation of HTML documents.</li> <li>3. Integrate and use external libraries such as Fontawesome for enhanced visual appeal.</li> <li>4. Fetch and display real-time weather data using APIs.</li> <li>5. Implement a feature to toggle between Celsius and Fahrenheit temperature units.</li> </ol>	<ol style="list-style-type: none"> <li>1. Develop a Weather Web App using HTML, CSS, and JavaScript.</li> <li>2. Integrate jQuery and Fontawesome libraries into a web project.</li> <li>3. Construct HTML structure to display weather data.</li> <li>4. Style the web app using CSS for an engaging user interface.</li> <li>5. Fetch and display real-time weather data from an API using JavaScript.</li> </ol>

## Week 7

### Lesson: Web Showcase

● Advanced

🕒 60 mins

👥 Teacher/Student led

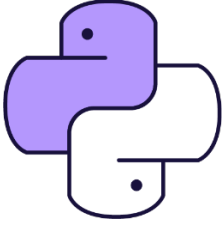
Prepare for a hands-on session where students will brainstorm, design, and code their own web page. Encourage creativity in content and design, emphasising the importance of HTML structure, CSS styling, and JavaScript interactivity. Guide them through project setup, content addition, styling, and refining their work. Ensure they test their work across different screen sizes and browsers, and foster a collaborative environment for feedback sharing.

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

- Chromebook/Laptop/PC
- iPad/Tablet

Learning Goals	Learning Outcomes
<ol style="list-style-type: none"> <li>1. Generate creative and unique ideas for a web page design and content.</li> <li>2. Establish a basic HTML structure for a web project using Codepen.</li> <li>3. Implement diverse HTML elements to structure and add content to a web page.</li> <li>4. Apply CSS for styling, enhancing visual appeal and readability of the web page.</li> <li>5. Integrate JavaScript to add interactivity to the web page and ensure its functionality.</li> <li>6. Review, refine, and optimise the web page for different screen sizes and browsers, while seeking and incorporating feedback.</li> </ol>	<ol style="list-style-type: none"> <li>1. Generate and articulate creative ideas for a web page design.</li> <li>2. Establish a web project using Codepen, incorporating the basic structure of an HTML document.</li> <li>3. Implement HTML to structure and add diverse content to a web page.</li> <li>4. Apply CSS to enhance the visual appeal of the web page, experimenting with colours, fonts, and layouts.</li> <li>5. Integrate JavaScript to add interactivity to the web page, ensuring functionality through thorough testing.</li> <li>6. Evaluate and refine the web page, ensuring code cleanliness, cross-browser compatibility, and responsiveness, and seek peer feedback.</li> </ol>

# Module: Introduction to Python







This module provides an introduction to Python programming, starting with basic syntax and the Microbit Python editor. Teachers should guide students through setting up their first project, creating a simple program, and introducing code sequence. The module progresses to cover variables, loops, conditional statements, operators, arrays, and functions. Each module includes a practical project to reinforce learning. The module culminates in a final project where students apply their skills to create a unique MicroPython project. Teachers should encourage experimentation and provide regular feedback.

Duration	Equipment
10 weeks	Students can use any of these devices: <ul style="list-style-type: none"> <li>• Chromebook/Laptop/PC</li> </ul> Required Equipment: <ul style="list-style-type: none"> <li>• Microbit</li> </ul>
Module Goals	Module Outcomes
<ol style="list-style-type: none"> <li>1. Understand and apply basic Python syntax and programming concepts using the Micro:bit Python editor.</li> <li>2. Master the use of variables, including declaration, assignment, and manipulation in Python.</li> <li>3. Comprehend and implement different types of loops and conditional statements in Python programming.</li> <li>4. Learn about and apply comparison operators, logical operators, and conditional Booleans in Python.</li> <li>5. Gain proficiency in working with arrays, including creating, manipulating, and applying advanced array tactics in Python.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and apply basic Python syntax and use the Micro:bit Python editor to create simple programs.</li> <li>2. Declare, assign, and manipulate variables in Python, culminating in the creation of a higher or lower game.</li> <li>3. Understand and implement different types of loops in Python, including while loops, for loops, and nested loops, and apply these in a reaction time game.</li> <li>4. Use conditional statements in Python to make decisions in code and apply these concepts in a Dice Roller project.</li> <li>5. Understand and use comparison operators, logical operators, and conditional Booleans in Python, and apply these in a Temperature Indicator project.</li> <li>6. Work with arrays in Python, including creating, manipulating, and retrieving elements from a list, and apply these skills in an LED light pattern project.</li> <li>7. Perform advanced operations with arrays in Python, including sorting, finding the length of a list, and counting occurrences, and apply these in a strong password generator project.</li> <li>8. Understand the differences between procedures and functions in Python and apply this knowledge in a weather station project.</li> <li>9. Understand the distinctions between local and global variables, understand variable scope, and apply these concepts in a Micro:bit temperature logger project.</li> <li>10. Conceptualize, plan, and build a unique project using Python and the Micro:bit, applying all the skills and knowledge acquired throughout the course.</li> </ol>

# Week 1

## Lesson: An Introduction to Python

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

Prepare to introduce Python as a beginner-friendly programming language, highlighting its use in various fields. Familiarise yourself with the Micro:bit Python editor for practical application. Discuss Python's syntax, particularly the importance of indentation and comments. Guide students through writing their first Python program and adding comments for clarity. Explain the sequence of code execution using a simple program. Encourage further practice and exploration post-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
<ol style="list-style-type: none"> <li>1. Understand and apply the basics of Python programming language.</li> <li>2. Utilise the Micro:bit Python editor for code writing and testing.</li> <li>3. Comprehend and implement Python's indentation rules to define code blocks.</li> <li>4. Use Python comments for code explanation and documentation.</li> <li>5. Write, run, and debug simple Python programs using Micro:bit.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and explain the basics of Python programming language and its application in various fields.</li> <li>2. Access and navigate the Micro:bit Python editor for writing and testing Python code.</li> <li>3. Apply Python indentation rules to define code blocks correctly.</li> <li>4. Use Python comments to add notes and explanations to the code.</li> <li>5. Write, run, and debug a simple Python program using the Micro:bit Python editor.</li> </ol>

## Week 2

### Lesson: Mastering Variables

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

This lesson guides students through the process of mastering variables in Python. They will learn about variable declaration, assignment, types, and naming conventions. The lesson also includes practical exercises such as creating a higher or lower game using the Microbits Python editor. Teachers should ensure students understand the concept of variables, their types, and how to manipulate them. They should also facilitate the game creation exercise, helping students apply their knowledge in a practical context.

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
<ol style="list-style-type: none"> <li>1. Understand the concept of variables in Python, including their declaration, assignment, and types.</li> <li>2. Learn and apply good variable naming conventions in Python.</li> <li>3. Manipulate variable values through operations such as incrementing, decrementing, and string concatenation.</li> <li>4. Apply knowledge of variables in creating a simple higher or lower game using the Microbits Python editor.</li> <li>5. Gain familiarity with importing and using libraries in Python, specifically for game development.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and apply the concept of variables in Python, including declaration, assignment, and types.</li> <li>2. Manipulate variable values through incrementing, decrementing, and string concatenation.</li> <li>3. Adhere to good naming conventions for variables, specifically snake_case and camelCase.</li> <li>4. Import and utilise libraries in Python, specifically for the creation of a higher or lower game.</li> <li>5. Develop a simple higher or lower game using variables, loops, conditionals, and libraries in Python.</li> </ol>

## Week 3

### Lesson: Looping Around

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

This lesson will guide students through understanding loops in Python, focusing on while loops, for loops, and nested loops. They will learn how Python uses indentation to define code blocks and how to break out of loops. The lesson includes practical exercises to reinforce learning, such as creating a project to measure reaction times. Teachers should ensure students understand the importance of consistent indentation and how loops can be used to control the flow of a program.

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
<ul style="list-style-type: none"> <li>• Understand the importance of indentations in Python and how they define code blocks.</li> <li>• Learn how to use 'while' loops and 'for' loops to repeat code execution based on conditions.</li> <li>• Explore nested loops and how they can be used to iterate through multiple dimensions.</li> <li>• Learn how to break out of loops using the 'break' statement when a specific condition is met.</li> <li>• Apply the knowledge of loops and control structures to create a simple reaction time game.</li> <li>• Develop problem-solving skills by modifying and extending the provided code examples.</li> </ul>	<ul style="list-style-type: none"> <li>• By the end of this lesson, students will be able to identify and differentiate between while loops, for loops, and nested loops in Python.</li> <li>• Students will be able to demonstrate the use of proper indentation in Python code to define code blocks.</li> <li>• Students will be able to create and manipulate while loops and for loops to execute a block of code multiple times.</li> <li>• Students will be able to implement nested loops to control the flow of their program through multiple levels of iteration.</li> <li>• Students will be able to use the 'break' statement to exit a loop prematurely based on a specific condition.</li> <li>• Students will be able to apply their knowledge of loops and control structures to create a simple reaction time game using the micro:bit's LED matrix and buttons.</li> </ul>

## Week 4

### Lesson: Making Decisions

● Intermediate

🕒 60 mins

👥 Teacher/Student led

📝 Student Quiz

💡 Student Challenge

Prepare to guide students through understanding conditional statements in MicroPython, including 'if', 'elif', and 'else'. They will create a simple project to reinforce their understanding, and then apply these concepts to a Dice Roller project. Ensure students understand how to use the Microbit Python Editor, and are comfortable with concepts such as loops, conditions, and using the micro:bit's accelerometer. Encourage experimentation with different conditions and scenarios to deepen their understanding.

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
<ol style="list-style-type: none"> <li>1. Comprehend and apply conditional statements in coding, including 'if', 'elif', and 'else'.</li> <li>2. Develop a basic project using 'if' statements to demonstrate understanding.</li> <li>3. Understand and implement 'elif' and 'else' statements to create complex decision-making structures.</li> <li>4. Grasp the concept of nested 'if' statements and their application in coding.</li> <li>5. Create a Dice Roller project utilising 'if', 'elif', and 'else' statements, demonstrating the ability to make decisions in code based on specific conditions.</li> </ol>	<ol style="list-style-type: none"> <li>1. Apply conditional statements in Python code, specifically if, elif, and else statements.</li> <li>2. Construct a simple if statement to check a condition and execute a block of code.</li> <li>3. Create complex decision-making structures using elif and else statements in conjunction with if statements.</li> <li>4. Utilise nested if statements to check multiple conditions within a single if statement.</li> <li>5. Develop a Dice Roller project using the micro:bit's accelerometer, random number generation, and conditional statements to display different outcomes.</li> </ol>



## Week 5

### Lesson: Operators Decoded

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

Prepare to guide students through understanding comparison and logical operators, and conditional Booleans in MicroPython. They'll apply these concepts in a practical project, creating a temperature indicator using the Microbit's online editor. Ensure they understand how to use these operators in 'if' and 'elif' statements. Encourage experimentation with different values to see how it affects conditions. The final project will involve using comparison and logical operators to determine temperature ranges and display appropriate images.

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
<ol style="list-style-type: none"> <li>1. Understand and apply comparison operators in MicroPython.</li> <li>2. Understand and apply logical operators in MicroPython.</li> <li>3. Understand and utilise conditional Booleans in MicroPython.</li> <li>4. Create a temperature indicator project using MicroPython and Micro:bit.</li> <li>5. Apply knowledge of operators and conditional Booleans in practical coding scenarios.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and apply comparison operators in MicroPython.</li> <li>2. Utilise logical operators to combine conditional statements in MicroPython.</li> <li>3. Implement conditional Booleans to make decisions based on the result of a condition.</li> <li>4. Create a temperature indicator project using the built-in temperature sensor of Micro:bit.</li> <li>5. Apply comparison and logical operators to determine temperature range and display appropriate images on Micro:bit.</li> </ol>

## Week 6

### Lesson: Array Essentials

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

In this lesson, teachers will guide students through the basics of working with arrays in MicroPython, using the micro:bit Python editor. Students will learn what an array is, how to create a list, retrieve and change list elements, add and remove elements from a list. The lesson culminates in a project where students will use arrays to create patterns of lights on the micro:bit LED display. Teachers should ensure students understand each step before moving on to the next.

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
<ol style="list-style-type: none"> <li>1. Understand and apply the concept of arrays in MicroPython.</li> <li>2. Create, retrieve, and modify elements in a list.</li> <li>3. Add and remove elements from a list.</li> <li>4. Use arrays to create patterns of lights on the micro:bit LED display.</li> <li>5. Combine and manipulate multiple arrays to create complex data structures.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and define arrays in MicroPython.</li> <li>2. Create, retrieve, and manipulate elements in a list.</li> <li>3. Add and remove elements from a list using <code>append()</code>, <code>extend()</code>, <code>remove()</code>, and <code>pop()</code> methods.</li> <li>4. Use arrays to store and manage data in MicroPython programs.</li> <li>5. Apply array manipulation skills to create an LED light pattern project on a micro:bit.</li> </ol>

## Week 7

### Lesson: Advanced Array Tactics

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

Prepare to guide students through advanced operations on Python lists using MicroPython. The lesson covers sorting lists in ascending and descending order, finding the length of a list, counting occurrences in lists, and applying these skills to create a strong password generator. Ensure students understand the use of `sort()`, `len()`, and `count()` methods, and how to use loops and `random.choice()` function. Encourage them to experiment with the code in their own projects.

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
<ol style="list-style-type: none"> <li>1. Master advanced operations on lists in Python, including sorting in ascending and descending order.</li> <li>2. Understand how to determine the length of a list using the <code>len()</code> function.</li> <li>3. Learn to count the occurrences of a specific item in a list using the <code>count()</code> method.</li> <li>4. Apply the learned concepts in a practical project to create a strong password generator.</li> <li>5. Develop skills in manipulating and analysing data stored in lists.</li> </ol>	<ol style="list-style-type: none"> <li>1. Sort a list in ascending order using the <code>sort()</code> method in MicroPython.</li> <li>2. Sort a list in descending order using the <code>sort(reverse=True)</code> method in MicroPython.</li> <li>3. Determine the length of a list using the <code>len()</code> function in MicroPython.</li> <li>4. Count occurrences of a specific item in a list using the <code>count()</code> method in MicroPython.</li> <li>5. Generate a strong, random password using a combination of character sets in MicroPython.</li> </ol>

## Week 8

### Lesson: Function Junction

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

In this lesson, students will delve into Python programming, focusing on procedures and functions. They will learn the difference between the two, create simple procedures and functions using the micro:bit, and understand the use of parameters in functions. The lesson culminates in a project where students design a simplified weather station using their new skills. This hands-on approach will help reinforce their understanding of procedures and functions in Python.

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
<ol style="list-style-type: none"> <li>1. Understand the difference between procedures and functions in Python programming.</li> <li>2. Create and utilise procedures in Python code.</li> <li>3. Create and utilise functions in Python code, including those that return values.</li> <li>4. Develop functions with parameters to enhance flexibility and functionality.</li> <li>5. Apply knowledge of procedures and functions to create a simple weather station project using MicroPython and Micro:bit.</li> </ol>	<ol style="list-style-type: none"> <li>1. Differentiate between procedures and functions in Python programming.</li> <li>2. Create and utilise a procedure in Python to display a smiley face on the micro:bit's display.</li> <li>3. Develop a function in Python that returns the square of a number and display the result on the micro:bit.</li> <li>4. Construct a function with parameters in Python that takes two numbers and returns their sum, displaying the result on the micro:bit.</li> <li>5. Design and implement a simplified weather station on the micro:bit using procedures and functions.</li> </ol>

## Week 9

### Lesson: Scope Showdown: Local vs. Global

 Expert	 60 mins	 Teacher/Student led	 Student Quiz	 Student Challenge
---	---	---	--	---

Prepare to guide students through understanding the concept of local and global variables in programming. Start with an introduction to the term 'scope', followed by a detailed explanation of local variables using Python code. Then, introduce global variables and their usage. Discuss best practices for using global variables. The lesson culminates in a practical project where students create a temperature logger using MicroPython, applying their understanding of local and global variables. Finally, wrap up the lesson by reinforcing the importance of variable scope in coding projects.

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
<ol style="list-style-type: none"> <li>1. Understand the concept of variable scope in programming.</li> <li>2. Distinguish between local and global variables and their usage.</li> <li>3. Apply the concept of local and global variables in Python programming.</li> <li>4. Adhere to best practices when using global variables.</li> <li>5. Develop a Micro:bit temperature logger project using both local and global variables.</li> </ol>	<ol style="list-style-type: none"> <li>1. Understand and differentiate between local and global variables in programming.</li> <li>2. Identify the scope of a variable and its accessibility within a program.</li> <li>3. Apply the concept of local variables within a function, demonstrating their limited scope.</li> <li>4. Utilise global variables appropriately within a program, demonstrating their wider scope.</li> <li>5. Combine the use of local and global variables in a practical project, demonstrating understanding of best practices.</li> </ol>

## Week 10

### Lesson: Python Showcase

 Expert

 60 mins

 Teacher/Student led

Prepare to guide students through a creative process of conceptualising, planning, and building a MicroPython project. Encourage brainstorming, idea selection, and project proposal creation. Facilitate feedback sessions and assist in refining ideas. Support students during the coding process, ensuring they test their code regularly. Finally, help students prepare a presentation to demonstrate their project, followed by a reflection on their learning journey.

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
<ol style="list-style-type: none"> <li>1. Develop creative problem-solving skills through conceptualising and planning a MicroPython project.</li> <li>2. Enhance brainstorming abilities and apply previous knowledge to generate project ideas.</li> <li>3. Improve communication skills by presenting project proposals and seeking feedback.</li> <li>4. Strengthen coding skills through prototyping and building a MicroPython project.</li> <li>5. Reflect on the learning experience, identifying challenges faced and strategies used to overcome them.</li> </ol>	<ol style="list-style-type: none"> <li>1. Generate and refine project ideas related to MicroPython and Micro:bit.</li> <li>2. Develop a comprehensive project proposal including purpose, features, and required components.</li> <li>3. Seek and incorporate feedback to improve the project concept.</li> <li>4. Code, test, and debug a MicroPython project on the Micro:bit.</li> <li>5. Present the completed project, demonstrating its features and discussing the development process.</li> </ol>

© 2025 Coding Ireland. All rights reserved.

This learning plan and its contents are provided exclusively for use with the Digital Skills Curriculum and may not be reproduced, distributed, or shared without prior written permission from Coding Ireland. For more information, please visit [www.codingireland.ie](http://www.codingireland.ie).