

# Computing Scheme of Work Unit 3.1 –

## Coding – New from 2021



# Contents

Introduction .....	4
PRIMM.....	4
Levels of Scaffolded coding tasks .....	5
Medium Term Plan .....	6
Lesson 1 – Using Flowcharts.....	7
Aims .....	7
Success Criteria.....	7
Resources.....	7
Preparation .....	7
Activities .....	7
Lesson 2 – Using Timers .....	10
Aims .....	10
Success Criteria.....	10
Resources.....	10
Preparation .....	10
Activities .....	10
Lesson 3 – Using Repeat .....	13
Aim .....	13
Success Criteria.....	13
Resources.....	13
Preparation .....	13
Activities .....	13
Lesson 4 – Code, Test and Debug .....	16
Aims .....	16
Success Criteria.....	16
Resources.....	16
Preparation .....	16
Activities .....	16
Lessons 5 and 6 – Design and Make Interactive Scene .....	19
Aims .....	19
Success Criteria.....	19
Resources.....	19
Preparation .....	19

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



Lesson 5 Activities .....	20
Lesson 6 Activities .....	23
Appendix 1: Display Boards.....	25
Assessment Guidance .....	29



# Introduction

This unit consists of six lessons that assume children have followed the Coding Scheme of Work in Years 1 and 2. If most of the class have not, use the Coding Catch-Up unit instead of this unit.

Key coding vocabulary is shown in **bold** within the lesson plans, use these new words in context to help children understand the meaning of them and start to build up, their vocabulary of coding words.

The Gibbon guided activities provide further practice of the concepts that the children will be learning and can be used as extension activities. More able children can be encouraged to explore other things that they can change in their programs and experiment with the options available, such as timers and 'if' statements.

Children will often be able to solve their own problems when they get stuck, either by reading through their code again or by asking their peers; this models the way that coding work is really done. More able children can be encouraged to support their peers, if necessary, helping them to understand but without doing the work for them.

**Note:** To force links within this document to open in a new tab, right-click on the link then select 'Open link in new tab'

## PRIMM

The coding lessons in these units are structured around the **PRIMM** approach. The whole approach may take place during a lesson or series of lessons.

**Predict...** what this code will do

**Run...** the code to check your prediction

**Investigate...** trace through the code to see if you were correct

**Modify...** the code to add detail, change actions/outcome

**Make...** a new program that uses the same ideas in a different way. Get creative!

Often lessons will start by looking at existing code, asking the children to 'read' it and make **Predictions** to what they think will happen when the code is run. You'll then **Run** the code and give them time to discuss what happens and relate it back to their predictions. You'll spend time with them **Investigating** the code, looking at how different parts work and helping them to understand how. Once children understand how the code works, they will be encouraged to **Modify** it - changing and adding code and re-running the program to view the impact of their changes. And once confident with this, they are encouraged to try and **make** their own program from scratch.


**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



## Levels of Scaffolded coding tasks

You can support children's learning and understanding by using different degrees of scaffolding when teaching children to code. The lessons provide many of these levels of scaffolding within them and using Free Code Chimp, Gibbon and Gorilla enables children to clarify their thinking and practise their skills. These are not progressive levels; children can benefit from all the levels of activities at whatever coding skill level they are:

Scaffolding	Task type	Examples of how to provide these opportunities
Most scaffolded  Least scaffolded	Copying code	By giving children examples of code to copy.
	Targeted tasks	<ul style="list-style-type: none"> <li>• Read and understand code</li> <li>• Remix code to achieve a particular outcome.</li> <li>• Debugging.</li> <li>• Use printed code snippets so that children can't run the code but must read it.</li> <li>• Include unplugged activities and 'explaining' tasks e.g. 'how do variables work?'</li> </ul>
	Shared coding	<ul style="list-style-type: none"> <li>• Sharing Challenge activities as a class or group on the whiteboard.</li> <li>• Complete guided activity challenges as a class.</li> <li>• After completing challenges; share methods to create a class version of the challenge.</li> <li>• Free coding as a class</li> </ul>
	Guided exploration	<ul style="list-style-type: none"> <li>• Exploring a limited repertoire of commands</li> <li>• Remixing code</li> <li>• Explore commands in free code before being taught what they do.</li> <li>• Use questioning to support children's learning.</li> <li>• PRIMM approach; Predict – Run – Investigate – Modify - Make</li> </ul>
	Project design and code	<b>Projects (imitate, innovate, invent, remix)</b> There are different ways to scaffold learning in projects. This process can be applied to programming projects; <ul style="list-style-type: none"> <li>• Using example projects e.g. the Guided 2Code activities.</li> <li>• Completing the challenges at the end of each guided activity.</li> <li>• Free code✓</li> <li>• Create a project that imitates a high-quality exemplar.</li> <li>• Remixing ideas.</li> <li>• Independently creating a brand-new program.</li> </ul>
	Tinkering	Use Free code Gorilla to access the full suite of 2Code objects and commands ✓  Use Free code to play and explore freely.

Adapted from work by Jane Waite - Computing at Schools <https://www.computingschool.org.uk/>

In Literacy, some teachers follow a progression that scaffolds learning to write texts. At first children read lots of examples of the genre of text they are going to create. Then they create an *imitation* of an example text. Next, they create a variation of the text (*remix and innovate*). Finally, they get to *inventing* a brand-new version.

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: @2simplesoftware



# Medium Term Plan

Lesson	Title	Success Criteria
<a href="#"><u>1</u></a>	Using Flowcharts	<ul style="list-style-type: none"> <li>Children can read and explain a <b>flowchart</b></li> <li>Children can use a <b>flowchart</b> to create a computer program.</li> <li>Children can create a computer program that uses <b>click events</b> and <b>timers</b>.</li> </ul>
<a href="#"><u>2</u></a>	Using Timers	<ul style="list-style-type: none"> <li>Children can create a program that uses a <b>timer-after command</b></li> <li>Children can create a program that uses a <b>timer-every command</b></li> <li>Children understand there can be different ways to solve a problem.</li> </ul>
<a href="#"><u>3</u></a>	Using Repeat	<ul style="list-style-type: none"> <li>Children understand how the <b>turtle object</b> moves.</li> <li>Children can use the <b>repeat command</b> with an <b>object</b>.</li> <li>Children can create a computer program that includes use of the <b>repeat command</b>.</li> </ul>
<a href="#"><u>4</u></a>	Code, Test and Debug	<ul style="list-style-type: none"> <li>Children can create computer programs using prior knowledge.</li> <li>Children can <b>run, test</b> and <b>debug</b> their programs.</li> <li>Children can consider <b>nesting</b> when <b>debugging</b> their programs.</li> </ul>
<a href="#"><u>5 &amp; 6</u></a>	Design and Make an Interactive Scene	<ul style="list-style-type: none"> <li>Children can use the <b>properties</b> table to set the <b>properties</b> of <b>objects</b>.</li> <li>Children can plan their <b>scene</b> and code before they create their program.</li> <li>Children can confidently make several different things happen in a program.</li> </ul>

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



# Lesson 1 – Using Flowcharts

## Aims

- To review previous coding knowledge.
- To understand what a **flowchart** is and how **flowcharts** are used in computer programming.

## Success Criteria

- Children can read and explain a **flowchart**.
- Children can use a **flowchart** to create a computer program.
- Children can create a computer program that uses **click events** and **timers**.

## Resources

Unless otherwise stated, all resources can be found on the [main unit 3.1 page](#). From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page.

- [Vocabulary Quiz Year 3](#).
- [Animal Character Flowchart](#)
- [Free Code Chimp](#) (this is found on the [main 2Code page](#)).
- (Optional) [Vocabulary flash cards](#). The teacher flash cards have been created so you can print them on A4 paper, cut them to size, fold them in half and glue them together. You can display and use these throughout coding lessons to support use of vocabulary.

## Preparation

- Set [Free Code Chimp](#) as a 2Do, call it 'Flowchart Program'.
- You could print and copy [Animal Character Flowchart](#) for children to refer to while coding.

## Activities

Introduction	<p>Display <b>slide 2</b> and outline the lesson aims.</p> <p>Display <b>slide 3</b> and outline the success criteria.</p>
Vocabulary	<p>Display <b>slide 4</b>, Use the Y3 Coding Vocabulary Quiz as a class to help refresh coding knowledge. It is set up so that you attempt all questions and then click the hand in button to check the answers. Click 'OK' to see which are correct and incorrect. Run through the</p>

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



	answers to the questions together. You could use the vocabulary cards to find the answers and display in the classroom.
Flowcharts	Display <b>slide 5</b> . Display the Animal and Character Flowchart. Tell children that this is a <b>flowchart</b> for a computer program. Give them some time to discuss with their talking partner what this program might do. Follow the activities on the slide.
Procedures	Display <b>slide 6</b> . Tell children that there are two parts to this <b>flowchart</b> – each part is called a <b>procedure</b> . Ask children to help you create this program in 2Code following the next slides.
Add a Background	Display <b>slide 7</b> . Open <a href="#">Free Code Chimp</a> and click on 'Design'. Ask children to remind you how to add the <b>background</b> . Choose a <b>background</b> with some land that the animal and character <b>objects</b> might move on.
Add Objects	Display <b>slide 8</b> . Add an <b>animal object</b> and a <b>character object</b> . (the <b>objects</b> they identified in the <b>flowchart</b> ).
Change Objects	Display <b>slide 9</b> . Show children how to change their objects by either double clicking on them or clicking on them and then clicking on the image in the <b>properties</b> table. Follow the guidelines on the slide.
Start Coding	Display <b>slide 10</b> . Click on 'Exit Design' to start adding the code. Recap <b>click events</b> while adding this code – the <b>when clicked event</b> triggers a <b>sound</b> , which is the <b>output</b> for this <b>event</b> .





Activity 1: Make Your Own Version	Display <b>slide 11</b> . Display the <b>flowchart</b> again and ask children to start <a href="#">Flowchart Program</a> from their 2Dos. Challenge them to create a scene and add code that implements both <b>procedures</b> in the <b>flowchart</b> . Their <b>background</b> and <b>objects</b> might be different, but their code should work in the same way.
How Did You Get On?	Display <b>slide 12</b> . Review the children's progress, share an example and look at the code together paying particular attention to fact that multiple <b>actions/ outputs</b> within an <b>event</b> are <b>indented</b> so it's clear that code will <b>execute</b> when that <b>event</b> happens.
Activity 2: Develop Your Program	Display <b>slide 13</b> . Challenge children to draw a <b>flowchart</b> for a third <b>procedure</b> on paper and then <b>develop</b> their program to include it. Ask children to hand in their 2Dos. Share an example of a finished <b>flowchart</b> and program and gain feedback from the children on how they got on.
Review Success Criteria	Display <b>slide 14</b> . Review the success criteria from <b>slide 3</b> . Children could rate how well they achieved this using a show of hands.



# Lesson 2 – Using Timers

## Aims

- To understand that there are different types of **timers**.
- To be able to select the right type of **timer** for a purpose.

## Success Criteria

- Children can create a program that uses a **timer-after command**.
- Children can create a program that uses a **timer-every command**.
- Children understand there can be different ways to solve a problem.

## Resources

Unless otherwise stated, all resources can be found on the [main unit 3.1 page](#). From here, click on the icon to set a resource as a 2do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page.

- [Magician](#). This is on the [main 2Code page](#) in the Chimp section.
- [Night and Day](#). This is on the [main 2Code page](#) in the Chimp section.
- [Tick Tock Challenge](#). This is on the [main 2Code page](#) in the Chimp section.
- [When Lightning Strikes Worksheet](#).

## Preparation

- Set [Night and Day](#) as a 2Do
- Set [Tick Tock Challenge](#) as a 2Do
- Print out copies of [When Lightning Strikes Worksheet](#).


## Activities

Vocabulary	Introduction	Display <b>slide 2</b> and outline the lesson aims.  Display <b>slide 3</b> and outline the success criteria.
	Vocabulary	Display <b>slide 4</b> . Display the key vocabulary <b>timer</b> on the board and recap knowledge from Year 2.
		Display <b>slide 5</b> . Display the key vocabulary <b>sequence</b> on the board and recap knowledge from Year 2.
		Display <b>slide 6</b> . Display the key vocabulary <b>timer</b> and <b>sequence</b> on the board and recap

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



	knowledge from Year 2.
Instructions With Delays	Display <b>slide 7</b> . Display this flowchart and ask the children to follow the instructions it gives. Ask children to draw their own version on an individual whiteboard or piece of paper. Ask children to swap their flowchart with a partner and have a go at following each other's instructions.
Magician	Display <b>slide 8</b> . Open <a href="#">Magician</a> and work through stages 1 – 4 together to recap children's understanding of how to use the <b>timer-after</b> code block. Children did this activity in Year 2; this is a recap. Make mistakes as you add the code and get the children to help you <b>debug</b> and fix the problems.
Magician – Stage 1	Display <b>slide 9. Stage 1</b> : This is a bit like using an <b>event</b> code block – it sets a <b>timer</b> and after the specified time the <b>object</b> (rabbit) will hide.
Magician – Stage 2	<p>Display <b>slide 10. Stage 2</b>: Add this incorrect code, <b>test</b> it and ask the children to help you <b>debug</b>:</p>  <p>When you <b>run</b> the program both <b>timers</b> start at the same time (if you click stop and play again you could notice they both highlight orange at the same time) and the code to 'hide' <i>and</i> 'show' the rabbit <b>executes</b> at the same time – after 5 seconds. Point out that the <b>timer</b> for the rabbit to 'show' needs to start <b>AFTER</b> the rabbit has hidden. You need to add the second <b>timer</b> <i>inside</i> the first <b>timer</b> OR work out that the rabbit 'shows' 10 seconds after the start (5 + 5) and alter the second <b>timer</b> to reflect that, so either of the solutions shown on the slide would work.</p>
Activity 1: Night and Day	Display <b>slide 11</b> . Ask children to start the <a href="#">Night and Day</a> activity from their 2Dos and see



	if they can complete it. This works in a very similar way to Magician.
Activity 2: Tick Tock Challenge	Display <b>slide 12</b> . Once they have completed Night and Day, and they have recapped using <b>timer-after</b> , tell children that there is another kind of <b>timer</b> , and they are going to learn about it by working through the Tick Tock Challenge. Set them to start and complete this challenge from their 2Dos. Review how they have got on – what have they learnt? Ask children: What is the difference between <b>timer-after</b> and a <b>timer-every</b> ?
Activity 3: Extension	Display <b>slide 13</b> . Hand out the When Lightning Strikes Worksheet and see if the children can read the code, <b>predict</b> what would happen and use the code to help them complete the <b>flowchart</b> . You could alternatively complete this extension task together on the board to finish the lesson.
Review Success Criteria	Display <b>slide 14</b> . Review the success criteria from <b>slide 3</b> . Children could rate how well they achieved this using a show of hands.



# Lesson 3 – Using Repeat

## Aim

- To understand how to use the **repeat command**.

## Success Criteria

- Children understand how the **turtle object** moves.
- Children can use the **repeat command** with an object.
- Children can create a computer program that includes use of the **repeat command**.

## Resources

Unless otherwise stated, all resources can be found on the [main unit 3.1 page](#). From here, click on the icon to set a resource as a 2do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page.

- [Free Code Chimp](#) (this is found on the [main 2Code page](#)).

## Preparation

- Set [Free Code Chimp](#) as a 2Do, call it 'Repeat Command'.

## Activities

Introduction	Display <b>slide 2</b> and outline the lesson aim. Display <b>slide 3</b> and outline the success criteria.
Vocabulary	Display <b>slide 4</b> . Introduce the new vocabulary and discuss the piece of coding shown.
The Repeat Command	Display <b>slide 5</b> . Explain to the children that they will be creating a program that uses the repeat command. Follow the instructions on the slide to start the scene.
Button Properties	Use <b>slide 6</b> to discuss and change the properties of the button, including text, size, text colour and background colour.
Draw a Square	Display <b>slide 7</b> and refer back to the aim of the task. Children to draw and think about the process of drawing a square. With <b>slide 8</b> , ask a child to walk in a square.

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



	Discuss the questions on the slide.
Program the Turtle	<p>Display <b>slide 9</b> and begin to program the turtle to draw a square.</p> <p>NB: If the children want to start with the button or when clicked, this is not incorrect, add that in and then add the repeat command inside the when clicked button event and continue.</p>
	<p>Display <b>slide 10</b>. Once the <b>repeat command</b> is set, we need to add inside it the <b>block of commands</b> that will repeat. Drag in the <b>object code block</b> for your turtle object (mySuperTurtle1 if you haven't renamed it) and look at the possible <b>actions</b> for that <b>object</b>. Look at the available actions and ask children what they want the turtle to do 4 times.</p>
	<p>Display <b>slide 11</b>. The children are likely to say 'forward', and then 'turn'. That is fine to start with – add in the code similar to that shown and then <b>test</b> it (if children have asked you to add the button in already the code for the repeat command will look like this, but it will be nested in a <b>when clicked button event</b>). You'll see when you <b>test</b> this code, that the turtle moves in a square but doesn't draw anything. Ask children to help you fix the problem – by adding in turtle 'pen down' and turtle 'pen up' <b>commands</b> at the beginning and end of your <b>block of commands</b>. <b>Test</b> the code, watch the turtle draw a square, continue to <b>test</b> and <b>debug</b> if needed.</p>
	<p>Display <b>slide 12</b>. Return to the aim of the task – ask children: Does our <b>algorithm</b> achieve this aim? We need to program all this to happen when the <b>button</b> is clicked on (unless the children have already told you to do program the button). Return to the code and ask children to help you add the <b>button</b> in and move (drag) the <b>repeat command</b> into the <b>when clicked</b> event for the <b>button</b>.</p>
Develop the Program	<p>Display <b>slide 13</b>. <b>Test</b> the code, click on the button, watch the turtle draw a square- or continue to <b>test</b> and <b>debug</b> if needed.</p>

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware





	Demonstrate to children how you could develop the program by altering the image and the background. Remind children that they can alter the image of the <b>turtle object</b> by double-clicking on it, or by clicking on the image in the <b>properties</b> table to open the clipart picker. Remind children how to rename <b>objects</b> .
Activity 1: Create your Program	Introduce the activity with <b>slide 14</b> . Ask pupils to start 'Repeat Command' from their 2Dos and create a program that uses a <b>turtle object</b> and the <b>repeat command</b> to draw shapes. They can choose any <b>background</b> and change their <b>turtle object</b> into something else from the clipart gallery.
Activity 2: Extension	Display an extension activity on <b>slide 15</b> . Ask children to see if they can develop their program by adding more objects and buttons, and explore what happens if they change the number of times the commands repeat or the angles the objects turn? Can they find out the number of sides and angles they'd need to set to make different shapes? – You could ask them to look it up or display them on the board for them to refer to.
Review Success Criteria	Display <b>slide 16</b> . Review the success criteria from <b>slide 3</b> . Children could rate how well they achieved this using a show of hands.



# Lesson 4 – Code, Test and Debug

## Aims

- To use coding knowledge to create a range of programs.
- To understand the importance of **nesting**.

## Success Criteria

- Children can create computer programs using prior knowledge.
- Children can **run**, **test** and **debug** their programs.
- Children can consider **nesting** when **debugging** their programs.

## Resources

Unless otherwise stated, all resources can be found on the [main unit 3.1 page](#). From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page.

- [Jumping Monkey](#). This is on the [main 2Code page](#) in the Chimp section.
- [Superheroes](#). This is on the [main 2Code page](#) in the Chimp section.
- [Rockets](#). This is on the [main 2Code page](#) in the Chimp section.

## Preparation

- Set [Jumping Monkey](#) as a 2Do for your class. This is on the [main 2Code page](#) in the Chimp section.
- Set [Superheroes](#) as a 2Do for your class. This is on the [main 2Code page](#) in the Chimp section.
- Set [Rockets](#) as a 2Do for your class. This is on the [main 2Code page](#) in the Chimp section.

## Activities

Introduction	<p>Display <b>slide 2</b> and outline the lesson aim.</p> <p>Display <b>slide 3</b> and outline the success criteria.</p>
Code, Test, Debug	<p>Display <b>slide 4</b>. Explain to children that in this lesson they will practise some of the programming skills they have learnt so far by using some guided lesson activities. Ask them to remind you of some of the things they have learnt about in coding lessons, these should include – <b>objects</b>, different types of <b>actions</b></p>

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)





	and <b>outputs, events, timers</b> , adding <b>sound</b> , using <b>repeat</b> , different types of <b>objects</b> , planning <b>algorithms, buttons, debugging, flowcharts</b> .
Vocabulary - Nesting	Display <b>slide 5</b> . Explain to children that as they get better at coding, they will want to make their programs more complex. It is important that they understand how to organise code. Ask them what they think <b>nesting</b> is in computer programming, and then discuss/ demonstrate what you mean when you talk about <b>nesting</b> in coding lessons.
	Display <b>slide 6</b> . Discuss further the definition of Nesting.
Code, Test, Debug	Display <b>slide 7</b> . Introduce the guided lessons that the children will be working through in today's lesson.  <b>NOTE</b> Children will need to <b>run, test and debug</b> their code in all of these activities. They must learn to think about how their code is organised – for example, is their timer inside the click event so that it doesn't start until the object is clicked on?
Jumping Monkey: Stage 1 and 2	Display <b>slide 8</b> . Before the children work through the lessons, run through the first three steps of Jumping Monkey making lots of mistakes, then <b>testing, debugging</b> and asking for the children's help to fix the problems. Recreate the code on the slide. It is important that the <b>timer</b> is inside the click event or it will start when you press play and the monkey will go down after 1 second, even if you haven't clicked on him!



Jumping Monkey: Stage 3	Recreate the code shown on <b>slide 9</b> in stage 3. In code example 1, the second <b>timer</b> you've added will start as soon as the program starts. In code example 2, both <b>timers</b> will start when the monkey is clicked (so the <b>commands</b> to go down and to stop will try and <b>execute</b> at the same time). Code example 3 is correct – the second timer needs to be <b>nested</b> inside the first timer so it starts when the first timer has finished.
Tips!	Display <b>slide 10</b> to go through some tips for when the children complete the activities independently.
Activity 1: Have a Go	With <b>slide 11</b> , instruct children to start the activities from their 2Dos.  Ask children to hand in their 2Dos when they have finished them and comment on how they got on.
Review Success Criteria	Display <b>slide 12</b> . Review the success criteria from <b>slide 3</b> . Children could rate how well they achieved this using a show of hands.



# Lessons 5 and 6 – Design and Make Interactive Scene

## Aims

- To design and create an interactive scene.

## Success Criteria

- Children can use the **properties** table to set the **properties** of **objects**.
- Children can plan their **scene** and **algorithms** before they create their program.
- Children can confidently make several different things happen in a program.

## Resources

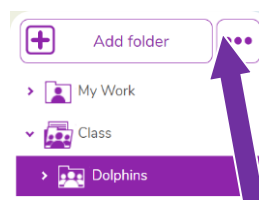
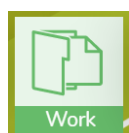
Unless otherwise stated, all resources can be found on the [main unit 3.1 page](#). From here, click on the icon to set a resource as a 2do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page.

- [Lightning Scene.](#)
- [Moon Phases.](#)
- [Solar System.](#)
- [Viking Discovery.](#)
- [Unicorn Dog Seagull.](#)
- [Free Code Chimp](#) (this is found on the [main 2Code page](#)).
- [Storyboard Planner.](#)
- [Scene and Code Planner.](#)
- [Sketch Plan Example.](#)

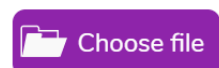
## Preparation

- Print and copy a range of planning documents for children to use in Lesson 5.
- Open the example programs: [Lightning Scene](#), [Moon Phases](#), [Solar System](#) and [Viking Discovery](#) in 4 browser tab for easy access.
- Set [Free Code Chimp](#) for children to refer to in Lesson 5 and use in Lesson 6.

Children might want to be able to use images that are photographs or not part of Purple Mash when creating their program. You could create a folder of topic-related images in the class folder for them to choose from when they are coding. You can upload these **in the work area** and then children can select 'choose file' from the galleries in 2Code to access them:



Children:



**Need more support? Contact**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)

**2simple**



Children could alternatively source them, save them and upload them to their own folder or import them directly from their device.

## Lesson 5 Activities

Introduction	<p>Display <b>slide 2</b> and outline the lesson aim.</p> <p>Display <b>slide 3</b> and outline the success criteria.</p>
Design and Make an Interactive Scene	<p>Display <b>slide 4</b> and introduce the main activity which will be spread over two lessons. Explain that the main aim of this lesson is to create a plan that would be good enough for someone else to follow to make your program if you didn't.</p>
1. Lightning Scene	<p>Display <b>slide 5</b>. Open the program, look at the design and the code and then run the program and discover how it works.</p> <p>Give children time to discuss what they like or don't like about each program, and how they might <b>develop</b> it – if they think it needs <b>developing</b>.</p> <p>Leave the program open in tabs.</p>
2. Moon Phases	<p>Display <b>slide 6</b>. Open the program, look at the design and the code and then run the program and discover how it works.</p> <p>Give children time to discuss what they like or don't like about each program, and how they might <b>develop</b> it – if they think it needs <b>developing</b>.</p> <p>Leave the program open in tabs.</p>
3. Solar System	<p>Display <b>slide 7</b>. Open the program, look at the design and the code and then run the program and discover how it works.</p> <p>Give children time to discuss what they like or don't like about each program, and how they might <b>develop</b> it – if they think it needs <b>developing</b>.</p> <p>Leave the program open in tabs.</p>

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



4. Viking Discovery	<p>Display <b>slide 8</b>. Open the program, look at the design and the code and then run the program and discover how it works.</p> <p>Give children time to discuss what they like or don't like about each program, and how they might <b>develop</b> it – if they think it needs <b>developing</b>.</p> <p>Leave the program open in tabs.</p>
Object Properties	<p>Display <b>slide 9</b> and look at the object properties. Open up the program Unicorn Dog Seagull. Talk through the options in the <b>properties</b> table row by row with the children. How does changing each <b>property</b> affect the <b>object</b>? In the 'Lightning Scene' most of the <b>objects</b> in the design were hidden, and they were programmed to show when they were needed the <b>sequence</b>.</p> <p>Set a sensible name for the objects in this scene. If 'allow off screen' is set to 'yes', the <b>object</b> will be allowed to move off the screen in the direction it's moving and it will disappear. If it is set to 'no' it will move off the screen in the direction it's moving, but come back on the other side, moving in the same direction. You could add a simple 'animal right' command to demonstrate this.</p>
Different Objects	<p>Display <b>slide 10</b>. Remind children that if there isn't an <b>object</b> type that matches what they want to add (e.g. they want to add a car) they can choose one that is there (e.g. animal) and then change the image and the name. If you return to 'Lightning Scene' and click on the fire engine you'll see in the properties table that the fire engine is actually an <b>animal object</b> with the image and name changing it to a fire engine.</p>
Show/Hide – Don't Forget Nesting!	<p>Display <b>slide 11</b> and look again at Unicorn Dog Seagull. Set the <b>show/hide property</b> for the seagull to hide and then add code to make it show after 3 seconds and say 'Hi!' when it shows – following last lesson, emphasise the importance of <b>nesting</b> the <b>speak command</b></p>



	into the <b>after timer</b> or the seagull will speak but you won't be able to see it!
Button Properties	Display <b>slide 12</b> . Drag a button onto the <b>scene</b> and remind children how the <b>properties</b> options are different.
Alert!	With <b>slide 13</b> , go back to 'Moon Phases' and see how there are instructions at the start – look at the code to see that these were programmed using an <b>alert</b> .
Planning	With <b>slide 14</b> , show the children the planning frameworks you've printed and copied and encourage them to choose a method of planning that suits them or that they think suits their program. They might favour a scene sketch like the one on slide 14, they could use the Storyboard Planner or the Scene and Code Planner they used in Year 2.
	Display <b>slide 15</b> . Give children time to make their plan, recommend that they have <a href="#">Free Code Chimp</a> open in front of them. If they don't finish this lesson they will be able to carry on in the next, and if they finish they could start making their programs in <a href="#">Free Code Chimp</a> .



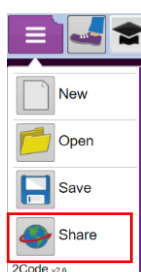
## Lesson 6 Activities

Introduction	Remind children from the last lesson:  Display <b>slide 2</b> and outline the lesson aim.  Display <b>slide 3</b> and outline the success criteria.
Planning	Display <b>slide 15</b> . Ask children to get out their program plans from last lesson and complete them if they haven't already.
Create Your Program	Display <b>slide 16</b> . When their plan is complete, ask the children to open <a href="#">Free Code Chimp</a> from their 2Dos and create their programs using their plans.
How Did You Get On?	Display <b>slide 17</b> . Remind children to save their 2Dos when they have completed it. Share children's work to a Displayboard (see <a href="#">Appendix 1</a> ) and allow them some time to view and interact with each other's interactive scenes. Review their work and celebrate achievements.
Review Success Criteria	Display <b>slide 18</b> . Review the success criteria from <b>slide 3</b> . Children could rate how well they achieved this using a show of hands.

If you want to share the programs children have created you can create a QR code or web link to them. This can be inserted into a school blog or webpage:

### How to Create a QR Code

- Save the file.
- From within the menu, click on 'Share':



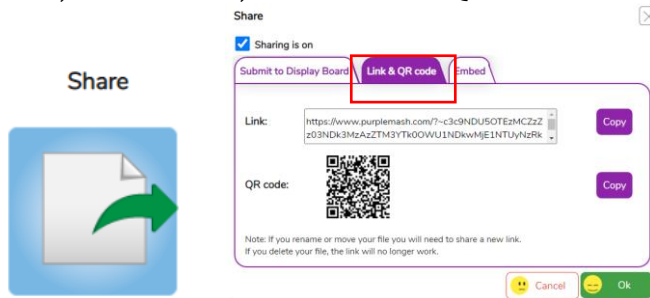
**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)





- Next, select Share, then Link and QR code



- The link and QR code can be copied and pasted into documents. Clicking on the QR code will show a large image that can be saved into the computer (right-click on it, choose Save As).



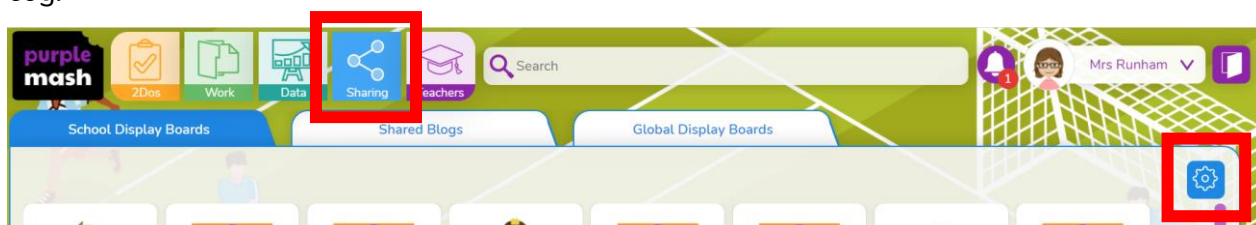


# Appendix 1: Display Boards

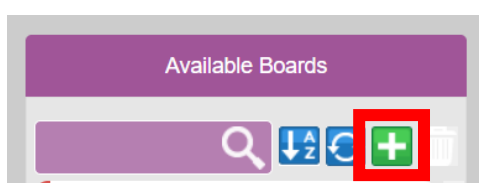
## Create the Display Board


Creating the display board is usually something you do before the lesson.

1. Click on the 'Sharing' button to find the Display Board tab, and then click on the settings cog:



2. Click on the '+' in the menu on the left:



3. Edit the settings (don't forget to add an icon by clicking on the ) , select the class and then click on 'Save':

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



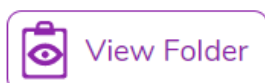
#### 4. Exit Display Board settings:



The Display Board will now be visible under the 'Sharing' button to all those you've selected to have access to it.

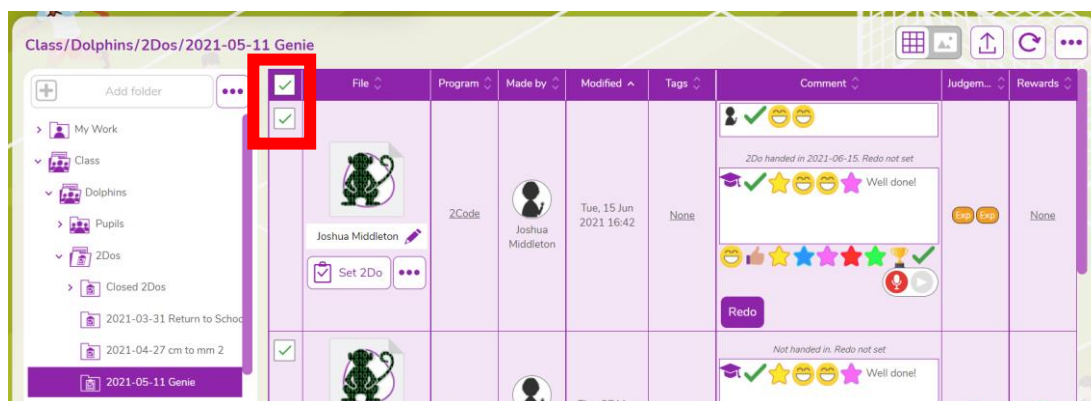
#### Adding work to a Display Board:

1. Click on 'View Folder' from the 2Do:



(or navigate to the work you want to share in the Work area).

2. Select the files you want to add to the display board or select all files in the folder using the tick at the top.

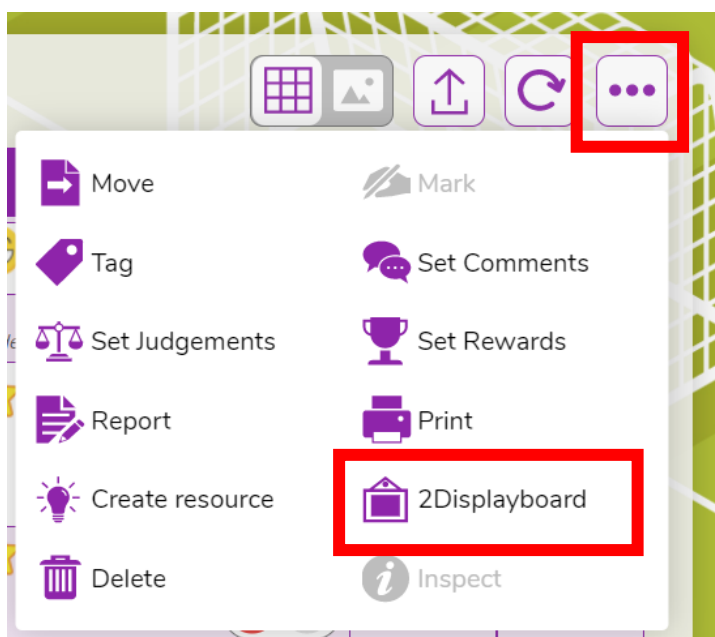


3. Click on the '...' menu button top right, then click on '2Displayboard':

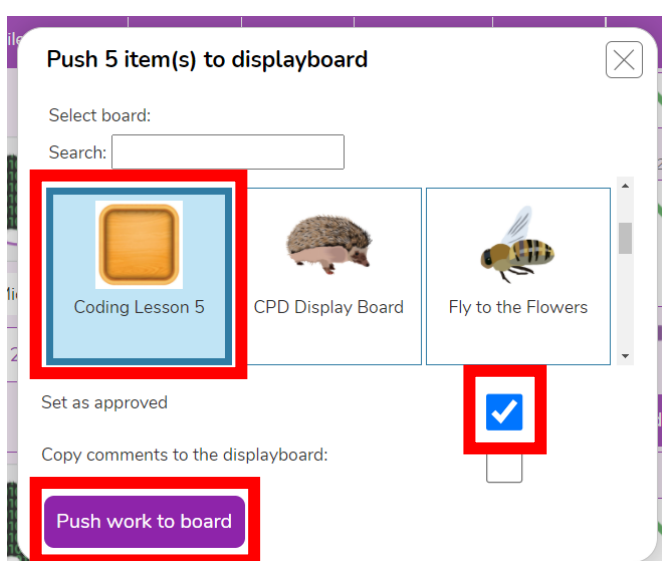
**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)





4. Choose the display board you've made for the work, tick 'Set as approved' and 'Push work to board':



5. Click on 'Sharing' button and then on the display board, you should see the work you've added. It can be deleted by clicking on 'Edit' at the top of the board, then clicking work and then delete. This will remove it from the display board, it won't delete it from Purple Mash.

### Deleting or Archiving a Display Board:

When you've finished the lesson, you can return to the Display board settings and either delete it or archive it to stop it appearing under the 'Sharing' button.

1. Click on 'Sharing' and then on the settings cog.

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



2. Tick 'Archive', and then 'Save' OR 'Delete'

*Clicking on 'Delete' will delete the display board but the work will still be available in the work area, it doesn't not delete the files.*

The screenshot shows the 'Coding Lesson 5' display board settings. The 'Name' and 'Description' fields are both set to 'Coding Lesson 5'. The 'Icon' is a purple square with a white 'P'. The 'Hide Info' section has 'Hide pupil name' checked and 'Hide class name' unchecked. The 'Access' section has 'Only staff can push' unchecked and 'Archived (hidden but still accessible with link)' checked. A red box highlights the 'Archived' checkbox. Below the 'Access' section is a 'View display board' button. The 'Who Can See' section shows 'All School' selected, with 'Classes' and 'Groups' listed below. A red box highlights the 'Save', 'Cancel', and 'Delete' buttons at the bottom of the form.

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



# Assessment Guidance

The unit overview for year 3 contains details of national curricula mapped to the Purple Mash Units. The following information is an exemplar of what a child at an expected level would be able to demonstrate when completing this unit with additional exemplars to demonstrate how this would vary for a child with emerging or exceeding achievements.

Assessment Guidance	
Emerging	<p>Children can design and code a program that follows a simple sequence (Unit 3.1 Lessons 1 and 2).</p> <p>Children can make good attempts to 'read' code and predict what will happen in a program which can help them to correct errors (Unit 3.1 Lessons 2 and 3).</p> <p>Children's designs for their programs, show that they are thinking of the structure of a simple program in logical, achievable steps (lessons 5 and 6).</p>
Expected	<p>Children have a clear idea of how to design and code a program that follows a simple sequence (Unit 3.1 Lessons 2 and 3). Children experiment with the use of timers to achieve delay effects in their programs – they understand the difference between timer-after and timer-every commands. (Unit 3.1 Lesson 2)</p> <p>Children' designs for their programs, show that they are thinking of the structure of a simple program in logical, achievable steps with attention to specific events that initiate specific actions. (Unit 3.1 Lessons 5 &amp; 6).</p> <p>Most children can explain the choice of commands they have included in their program and what they achieve (Unit 3.1 Lessons 5 &amp; 6).</p> <p>Children are able to use the repeat command to program a turtle to draw a square (Unit 3.1 Lesson 3)</p> <p>Children are beginning to understand how code is structured and are able to apply this knowledge when debugging (Unit 3.1 Lesson 4).</p> <p>Most children can integrate multimedia components such as sounds, animation and images into their coding. They can apply specific actions to these objects to animate them as part of the overall process of creating their own program (Unit 3.1. Lessons 5 and 6).</p> <p>They can be reflective on how successful they are at creating their programs and how the previous learning has helped them (Unit 3.1.).</p>
Exceeding	<p>Children's designs show that they are thinking of the required task and how to accomplish this in code (Unit 3.1 Lessons 5 &amp; 6).</p> <p>Children can identify an error within a program that prevents it following the desired algorithm and then fix it (Unit 3.1). Children make intuitive attempts to debug their own programs as they increase in complexity (Unit 3.1 Lesson 4).</p> <p>Children are able to use the repeat command to produce outcomes beyond the</p>

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)



## Assessment Guidance

set task (Unit 3.1 Lesson 3)

Children have a good understanding of timers within timers in a program (Unit 3.1 Lessons 2 and 4) and this is evidenced in their program designs (Unit 3.1 Lessons 5 & 6).

Children exhibit greater ease at fixing their own bugs as their coding becomes more complex. (Lessons 5 and 6).

**Need more support? Contact us:**

Tel: +44(0)208 203 1781 | Email: [support@2simple.com](mailto:support@2simple.com) | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)