DigiTech Scheme of Work Unit 5.1 – - New From 2023

Contents

Introduction

This unit consists of six lessons that assume children have followed the Coding Scheme of Work in Years 1 to 4. If most of the class have not, use the Coding Catch-Up unit instead of this unit.

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

The Gorilla 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 variables 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 thought 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 have an understanding of 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.

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	• Read and understand code • Remix code to achieve a particular outcome. • Debugging. • Use printed code snippets so that children can't run the code but must read it. • Include unplugged activities and 'explaining' tasks e.g. 'how do variables work?'
	Shared coding	• Sharing Challenge activities as a class or group on the whiteboard. • Complete guided activity challenges as a class. • After completing challenges; share methods to create a class version of the challenge. • Free coding as a class
	Guided exploration	• Exploring a limited repertoire of commands • Remixing code • Explore commands in free code before being taught what they do. • Use questioning to support children's learning. • PRIMM approach; Predict – Run – Investigate – Modify - Make
	Project design and code	**Projects (imitate, innovate, invent, remix)** There are different ways to scaffold learning in projects. This process can be applied to programming projects; • Using example projects e.g. the Guided 2Code activities. • Completing the challenges at the end of each guided activity. • Free code✓ • Create a project that imitates a high-quality exemplar. • Remixing ideas. • Independently creating a brand-new program.
	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.computingatschool.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.

Medium-Term Plan

Lesson	Title	Aims (Objectives)	Success Criteria
1	Coding Efficiently	• To review existing coding knowledge. • To begin to be able to simplify code. • To create a playable game.	• Children can use simplified code to make their programming more efficient. • Children can use variables in their code. • Children can create a simple playable game.
2	Simulating a Physical System	• To understand what a simulation is. • To program a simulation using 2Code.	• Children can plan an algorithm modelling the sequence of traffic lights. • Children can select the right images to reflect the simulation they are making. • Children can use their plan to program the simulation to work in 2Code.
3	Decomposition and Abstraction	• To know what decomposition and abstraction are in Computer Science. • To take a real-life situation, decompose it and think about the level of abstraction. • To use decomposition to make a plan of a real-life situation.	• Children can make good attempts to break down their task into smaller achievable steps. • Children recognise the need to start coding at a basic level of abstraction to remove superfluous details from their program that do not contribute to the aim of the task.
4	Friction and Functions	• To understand how to use friction in code. • To begin to understand what a function is and how functions work in code.	• Children can create a program which represents a physical system. • Children can create and use functions in their code to make their programming more efficient.
5	Introducing Strings	• To understand what the different variable types are and how they are used differently. • To understand how to create a string.	• Children can create and use strings in programming. • Children can set/change variable values appropriately. • Children know some ways that text variables can be used in coding.

6	Text Variables and Concatenation	• To begin to explore text variables when coding. • To understand what concatenation is and how it works.	• Children can create a string and use it in their program. • Children can use strings to produce a range of outputs in their program.

Lesson 1 – Coding Efficiently

Aims

- To review existing coding knowledge.
- To begin to simplify code.
- To create a playable game.

Success Criteria

- Children can use simplified code to make their programming more efficient.
- Children can use variables in their code.
- Children can create a simple playable game.

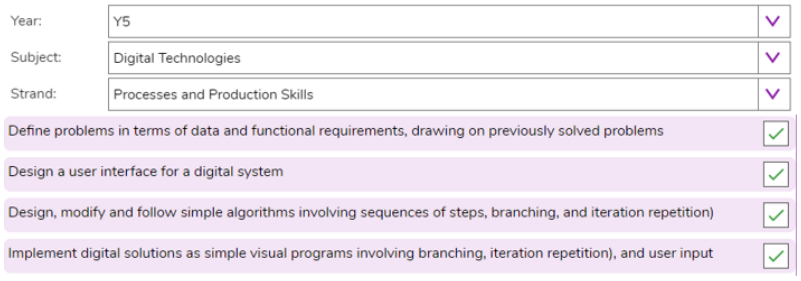
Resources

Unless otherwise stated, all resources can be found on the main unit 5.1 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 don't lose this page.

- Y5 Coding Vocabulary Quiz.
- Animal Race 1
- Animal Race 2
- Catching Game.
- Free Code Gorilla.
- 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 Catching Game as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:



- Create a display board for the class to share their programs to. Details of how to do this are given in Appendix 1

Activities

Introduction	Display slide 2 and outline the lesson aims. Display slide 3 and outline the success criteria.
Vocabulary	Display slide 4. Use the Y5 Coding Vocabulary Quiz as a class to help refresh coding knowledge from previous years. It is set up so that you attempt all questions and then click the  button to check the answers. Click 'OK' to see which are correct and incorrect: Run through the answers to the questions together. You could use the vocabulary cards to find the answers and display in the classroom or use slide 5 which has definitions.
	Slide 5 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
Activity 1: Animal Race	Display slide 6. Ask the children to look at the design and read the code, can they predict what will happen when the program is run? Use the slide to open Animal Race 1, click on play to run the program and click on the animals to see if their predictions were correct. Recap event – object – action, identifying each in this code.
	Display slide 7. The design in this program is the same, but the code is different. Can children predict what will happen when this program is run? Use the slide to open Animal Race 2, click on play to run the program and click on the animals to see if their predictions were correct. Explain to children that in this lesson they will revise some of the vocabulary and concepts they have learnt in Year 5, and start being able to simplify code to make their programming more efficient. Discuss what it might mean to make things more efficient. Return to slide 6 and begin to look at how this code works.

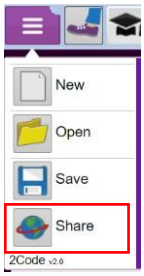
	Display slides 8-9. Click through the slides and use them to help you explain how the code in Animal Race 2 works. In year 5 children will use Free Code Gorilla. At the end of slide 9 you could open Free Code Gorilla at this point and use it to recap children's understanding of different object types.
Catching Game	Use slide 10. Display Catching Game on the board. The following slides go through and explain points about this activity. Complete it on the board together or get children to open it as well and complete it with you one step at a time using the following slides. Refer to key vocabulary and concepts as you go through it with the children including selection, IF/ELSE Statements, prompt, number variable, variable, timer (after/ every), event, object, action, co-ordinates.
Catching Game: Stage 1	Slide 11: Catching Game Stage 1 Start by clicking on 'Design' and looking at the scene together, there is a score, a catcher and some food objects. Click on the food objects and set a speed in the Attribute Value box for each one– around 1-3 is sensible!
Catching Game: Stages 2 and 3	Slide 12: Catching Game Stages 2 and 3 Stage 2: Create arrow key press events so the player can control the catcher. Stage 3: Add a collision detection event so that when the catcher collides with a weight, a sound plays and the game starts again. Explain that because there are 2 objects with the same tag (avoidables), 2Code gives you the option of using 'Any avoidable' to save you having to program a collision detection for each one. It has automatically generated a variable for this object type. NB weight is a custom object type that is not usually available in 2Code but was created for this example only. This puts into practise simplifying the code. When you test this code children might notice that the sound doesn't play because the program restarts straight away (re-run and watch when the code highlights orange to notice this) – discuss with them how that might be overcome later when we are able to fix it.

Catching Game: Stages 4 and 5	Slide 13: Catching Game Stages 4 and 5 Stage 4: Write code that increases the score by a value of 1 and plays a sound when the catcher collides with the food. Step 5: Write code so that the food hides when the catcher collides with it. The individual food object that needs to hide is the value of the 2Code generated variable, 'Any catchable'. 'Any catchable' is a variable generated by 2Code which could contain any of the food items – whichever is clicked on. This variable value will be set by the collision detection event, so they will need to use 'change variable' for this. Once they drag 'change variable' into the collision detection event they can select 'Any catchable'. Stage 6: Ask the children for ideas for how the game can be improved and together fix the problem of the sound not playing when the catcher bumps into a weight – add in a timer so the game restarts after a second.
Activity: Improve Catching Game	EITHER (Slide 14) Ask children to open Purple Mash and work through the Catching Game 2Do. They should work through all the steps you've been through and then try and improve the game in the final stage – challenge them to include x and y co-ordinates and an IF statement in their code. Challenge them to fix the problem where the sound doesn't play before the game restarts when the catcher collides with a weight. Remind children they can click on the instruction to bring the video back up. OR (Slide 15) Ask children to use Free Code Gorilla to have a go at creating their own game. OR both!
How did you get on?	Display slide 16. Share children's work 2Displayboard (see Appendix 1) and allow them some time to play each other's games. Review their work and celebrate achievements.
Vocabulary Overview	Slide 17 can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Display slide 18. Review the success criteria from slide 3. Children could rate how well they achieved this using a show of hands.

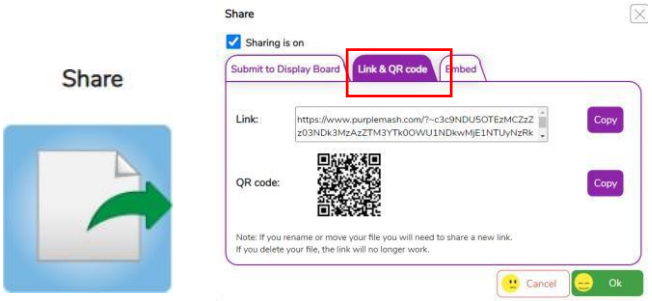
*If you want to share games you can create a QR code or web link to them. This can be inserted into a school blog or webpage:

How to Create a QR Code

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



- 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).

Remember to close your 2Dos when you have finished the lesson.

Lesson 2 – Simulating a Physical System

Aims

- To understand what a simulation is.
- To program a simulation using 2Code.

Success Criteria

- Children can plan an algorithm modelling the sequence of traffic lights.
- Children can select the right images to reflect the simulation they are making.
- Children can use their plan to program the simulation to work in 2Code.

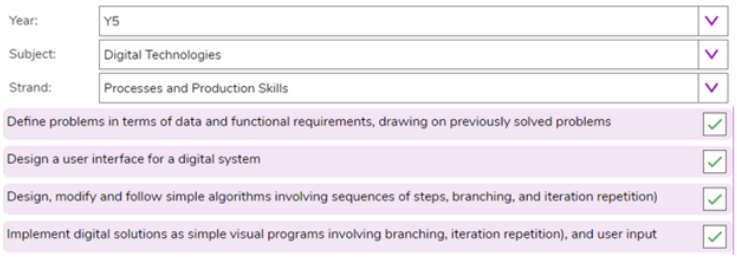
Resources

Unless otherwise stated, all resources can be found on the main unit 5.1 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 don't lose this page.

- Video of the UK traffic light sequence.
- Traffic Light Algorithm vocabulary.
- Traffic Light Algorithm writing frame.
- Traffic Light Sequence Flowchart.
- Free Code Gorilla.
- 2Chart

Preparation

- Open the video in browser tab.
- Print Traffic Light Algorithm Vocabulary and put it on display.
- Set the Traffic Light Algorithm writing frame OR Traffic Light Sequence Flowchart as a 2Do. You can select the following computing objectives when setting the 2Dos to make future assessment easier:



- Set Free Code Gorilla as a 2Do.
- You may wish to set the Traffic Lights Gibbon activity as a 2Do for children who may need a more straightforward/ simplified activity, whilst still accomplishing the same objectives.

Activities

Introduction	Display slide 2 and outline the lesson aims. Display slide 3 and outline the success criteria.
Vocabulary	Slide 4 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
	Use slide 5 to remind the children of the word algorithm and simulation.
Simulating a Physical System	Display slide 6. Use the slide to explain to children they will be writing an algorithm for a program that simulates a physical system.
Traffic Light Algorithm Vocabulary	Use slide 7 to share the vocabulary that could be included for a traffic light simulation or share the PDF.
Video of a UK Traffic Light Sequence	Use slide 8 to watch the Traffic Light Sequence video together. Ask the children to take notes that will enable them to write the algorithm. The algorithm is a sequence of instructions – discuss why it is so important in this case that the sequence is correct. You may need to play the video more than once.
Planning the Algorithm	Display slide 9. Give the children around 10 minutes to complete the following task (choose which one you set) *EITHER* Traffic Lights Flowchart: ask them to have a go at creating the flow chart for the traffic lights. *OR* Traffic Lights Algorithm: Ask them to use the writing template to formulate the algorithm for the traffic lights.

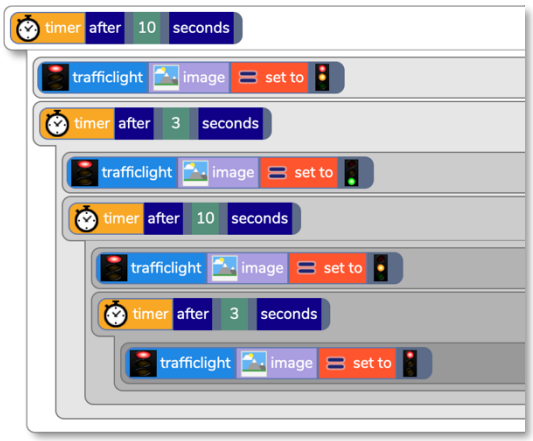
<table>
<tr><td></td><td>If using 2Chart show children how to change the text and add a title by double-clicking on the existing text. It should look something like these examples:

<table>
<tr><td>Turn on red</td></tr>
<tr><td>Wait 10 seconds</td></tr>
<tr><td>Turn on amber</td></tr>
<tr><td>Wait 3 seconds</td></tr>
<tr><td>Turn off red</td></tr>
<tr><td>Turn off amber</td></tr>
<tr><td>Turn on green</td></tr>
<tr><td>Wait 10 seconds</td></tr>
<tr><td>Turn on amber</td></tr>
<tr><td>Turn off green</td></tr>
<tr><td>Wait 3 seconds</td></tr>
<tr><td>Turn off amber</td></tr>
<tr><td>Repeat all forever</td></tr>
</table>

</td></tr>
<tr><td>Activity 1: Setting up the Scene</td><td>Display slide 10. Review how children got on and then reveal the slide to them with the first task of setting the scene. Children to open their 2Dos – Free Code Gorilla.</td></tr>
<tr><td>Activity 2: Making the Lights Change</td><td>Display slide 11. Children to look at planned algorithm to help them create code that make the lights changes.

They will need to ensure that the timers are all nested (inside each other).

Here is an example using the algorithm:


</td></tr>
<tr><td>Making the Sequence Repeat Forever</td><td>Reveal slide 12. Use the slide to get children thinking about how we get the sequence to repeat forever.</td></tr>
</table>

Activity 3: Making the Sequence Repeat Forever	Display slide 13. Children to have a go at making the sequence last forever.
Activity 4: Adapting your Traffic Light Sequence	Display slide 14. Share some of the simulations so far made with the class. Children to have a go at adapting their code for the scenarios on the slide.
Activity 5: Extension	Display slide 15. Children try to simulate a pedestrian crossing.
Vocabulary Overview	Slide 16 can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from slide 3. Children could rate how well they achieved this using a show of hands.

Remember to close your 2Dos when you have finished the lesson

Lesson 3 – Decomposition & Abstraction

Aim

- To know what decomposition and abstraction are in Computer Science.
- To take a real-life situation, decompose it and think about the level of abstraction.
- To use decomposition to make a plan of a real-life situation.

Success criteria

- Children can make good attempts to break down their task into smaller achievable steps.
- Children recognise the need to start coding at a basic level of abstraction to remove superfluous details from their program that do not contribute to the aim of the task.

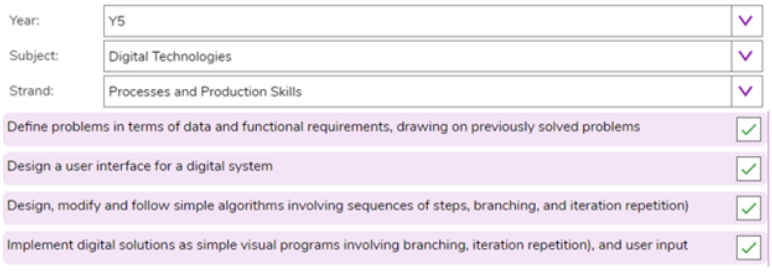
Resources

Unless otherwise stated, all resources can be found on the main unit 5.1 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.

- Some examples of simple board games such as snakes and ladders, chess, and solitaire.
- Football Example
- Decomposition and Abstraction writing frame

Preparation

- Set Decomposition and Abstraction writing frame as a 2Do for your class OR
- Print out the Decomposition and Abstraction writing frame, one for each child or pair of children. You can select the following computing objectives when setting the 2Dos to make future assessment easier:



Activities

Introduction	Display slide 2 and outline the lesson aims. Display slide 3 and outline the success criteria.
Vocabulary	Slide 4 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
	Display slide 5. Reveal slide to go through vocabulary and definitions of decomposition and abstraction.
Decomposition	Display slide 6. Use this slide to discuss decomposition and the example within it.
Football Game	Display slide 7. As a class, open the Football example and explore slide prompts together.
Abstraction	Display slide 8. Explore the Abstraction slide together.
Decomposition and Abstraction	Display slide 9. Explain to the class that, today, they are going to try these two processes – decomposition and abstraction – to design and write algorithms for either a simple board game (perhaps share some of the examples you have), or the scenario of planning and making a meal.
	Display slide 10. Explore the 'pie' example on the slide. Children could work in pairs to discuss and feedback to the class.
Activity 1: Decomposition and Abstraction	Use slide 11 to introduce the activity. Children to plan a process (meal, board game etc) use the writing frame: Decomposition and Abstraction.
How did you get on?	Display slide 12. Share the key questions on the slide.
Vocabulary Overview	Slide 13 can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from slide 3. Children could rate how well they achieved this using a show of hands.

Lesson 4 – Friction and Functions

Aims

- To understand how to use friction in code.
- To begin to understand what a function is and how functions work in code.

Success Criteria

- Children can create a program which represents a physical system.
- Children can create and use functions in their code to make their programming more efficient.

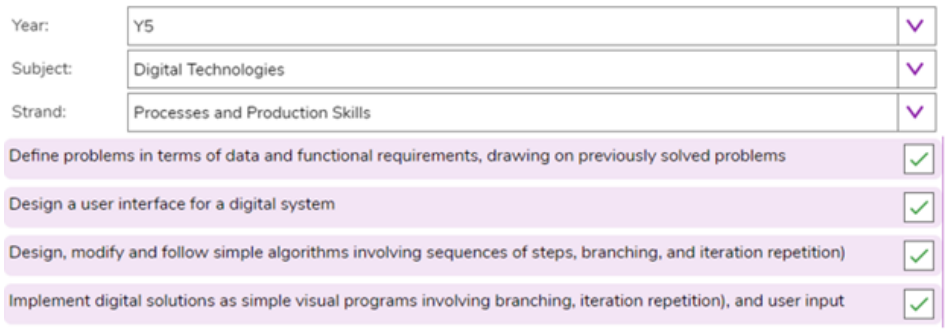
Resources

Unless otherwise stated, all resources can be found on the main unit 5.1 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 don't lose this page.

- Football Game. This is in the Gorilla activities.
- Friction Example.
- Physical Football (Optional)

Preparation

- Set Football Game as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:



- Create a display board for the class to share their programs to. Details of how to do this are given in Appendix 1.
- Physical Football (Optional)

Activities

Introduction	Display slide 2 and outline the lesson aims. Display slide 3 and outline the success criteria.
Vocabulary	Slide 4 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
Football	Remind children of the lesson where they made the simulation of a traffic light sequence – in that lesson they made a program that simulated a physical system. Explain that often when coding we want to simulate physical systems – program objects to behave in a realistic way. Gently kick a football across a space in front of the class. Watch it roll, then come to a stop. Ask children what would happen if you'd kicked it harder? And what if you were on a wooden floor and not carpet? Display slide 5. Explore this slide together as a class. Discuss the code and questions.
Football Friction	Display slide 6. Open the Friction Example. Go through the slide together as class. You might choose to let the children explore the Friction Example in small groups and feed back to the class.
	Display slide 7. Discuss with the children the code on the slide. Explore how friction has been set for each surface.
Football Game	Display slide 8. Children to open Football Game from their 2Dos. Explain the stages and what they are simulating. Display slides 9 to 11 respectively working from stages 1 to 4 as a class, modelling each of the stages (1 to 4). • Stage 1: Choose 'any' so the ball can be swiped in any direction. A speed of 1-4 is sensible. • Stage 2: Add friction to the ball, friction of between 1-4 is about right, test and change it and see what children want to set. • Stage 3: Involves altering existing code so click on what you want to change – the speed, click on the value for the current speed and select swipe speed so the speed of the ball will match the speed of the swipe. Drag in football and set the angle in the same way. • Stage 4: Introduces children to functions.

	• Watch the video together. • At the start of the program the ball is at X=3, Y=8 and the speed = 0. • Creating this function will apply those properties to the ball whenever it is used (called). • Use 'create function' to create the function – a function is a type of variable. This is a difficult concept to comprehend until you are creating more complex code (e.g. Java or Python coding) when the reasoning becomes clearer. • And then use 'call function' to add your function into your collision detection event.
Activity 1: Football Game	Display slide 12. Use this to set the children off on completing the Football Game from their 2Dos from start to finish. Children to carry on until they have made the football game. Review children's progress and ask them how many times they called their function. The function contained 3 pieces of code: - The X Co-ordinate - The Y Co-ordinate - The speed Notice together how calling the function saved them coding all this each time. Functions also help us simplify code and make our programming more efficient.
How did you get on?	Display slide 13. Share children's work on a Displayboard (see appendix 1) and give them some time to play each other's games.
Vocabulary Overview	Slide 14 can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from slide 3. Children could rate how well they achieved this using a show of hands.

Remember to close your 2Dos when you have finished the lesson.

Lessons 5 – Introducing Strings

Aims

- To understand what the different variable types are and how they are used differently.
- To understand how to create a string.

Success Criteria

- Children can create and use strings in programming.
- Children can set/change variable values appropriately.
- Children know some ways that text variables can be used in coding.

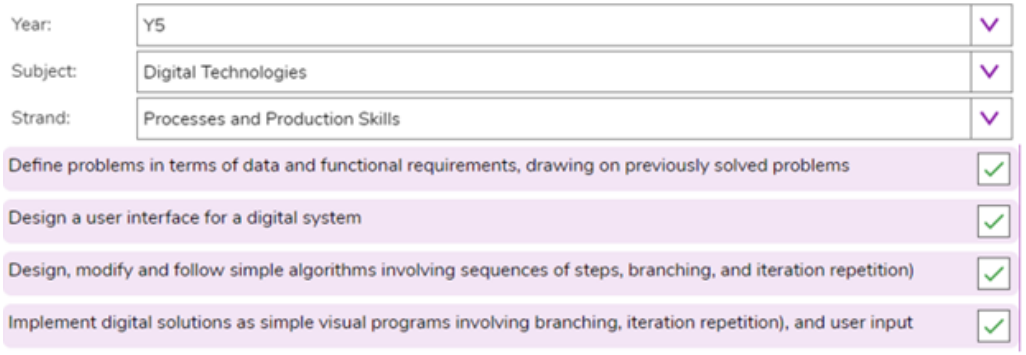
Resources

Unless otherwise stated, all resources can be found on the main unit 5.1 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 don't lose this page.

- Alien Blast Game.
- Alien Blast Scene.

Preparation

- Set Alien Blast Scene as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:



- Create a display board for the class to share their programs to. Details of how to do this are given in Appendix 1.

Activities

Introduction	Display slide 2 and outline the lesson aims. Display slide 3 and outline the success criteria.
Vocabulary	Slide 4 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
	Display slide 5. Go through the definition of variables on the slide and use it to remind the children of where they have encountered variables before.
Variables	Display slide 6. Use the slide to help children understand that you can select either: number, string or function.
Alien Blast	Display slide 7. Introduce strings by opening 'Alien Blast Game'. Click on 'Design' and explain that in this game the rocket blasts the aliens when it collides with them.
	Display slide 8. Click on 'See Code' and explain to children that you are going to look through the code tabs with them to see how it works. Start by looking at the 'Instruction Tab'. Instruction Tab The code in here will trigger an alert at the start to tell the player how to control the rocket.
	Display slide 9. Use the slide to review the 'Progress Tab' Progress Tab The progress tab contains code to keep the player informed of their progress. There is an initial message that will print to the screen at the start. A string is sequence of characters, which could form words, phrases or even whole sentences. There is a 'describe' string set as nothing. This string will be set to contain a word. Where will the value for this variable come from? What will it be? Explain that variables that are strings can also be called text variables.
	Display slide 10. Use the slide to continue to review the 'Progress Tab'. Read down to see that the value of 'describe' is set to a different random adjective every quarter second.

	These adjectives will be used to describe the aliens as they are blasted. • Click on the Run button to run the code. • Click on OK on the alert, then look for the 'Variable Watch' box. • Look for 'describe'. • Watch it change to a different random adjective every quarter second.
	Display slide 11. Click on the stop button and return to looking at the code: Collision The code in here programs what happens each time an alien is blasted. Where does it use the 'describe' string? What will this look like in the game?
	Display slide 12. Review the 'Rocket Tab'. Rocket The code in here will enable the player to control the rocket. It uses speed and angles. Play the game together as a class and explain that they will be making their own version of this game.
Activity 1: Alien Blast	Display slide 13. Ask children to open Alien Blast Scene (a starting point for the program you've just looked at – with a rocket, and alien and no code) from their 2Dos. Challenge children to use the 'clone' button in Design view to help create their scene, they should change and rename the objects, they might want to change the rocket. Check code for how the rocket moves – you might need to recap how angles are used. They must create and use a text variable and should also be encouraged to include a timer and a score.
How did you get on?	Display slide 14. Use this as an opportunity to share children's games to a display board (see appendix 1). Children could critique each other's games.
Vocabulary Overview	Slide 15 can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from slide 3. Children could rate how well they achieved this using a show of hands.

Remember to close your 2Dos when you have finished the lesson.

Lesson 6 – Text Variables and Concatenation

Aims

- To begin to explore text variables when coding.
- To understand what concatenation is and how it works.

Success Criteria

- Children can create a string and use it in their program.
- Children can use strings to produce a range of outputs in their program.

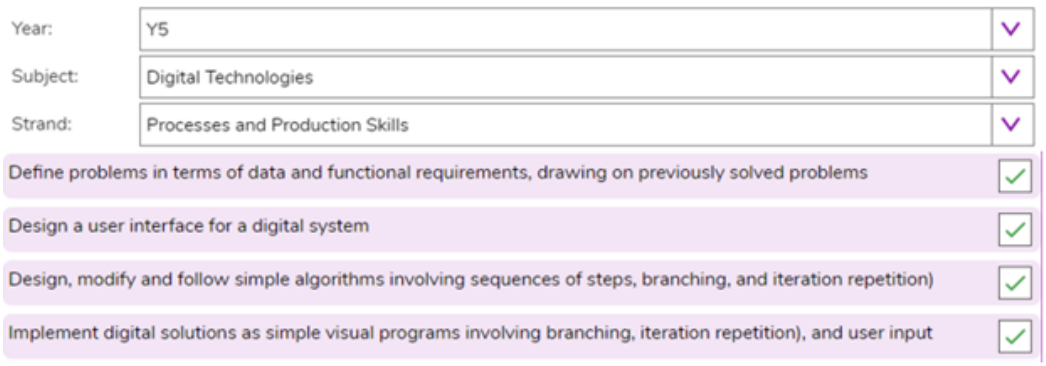
Resources

Unless otherwise stated, all resources can be found on the main unit 5.1 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 don't lose this page.

- Free Code Gorilla.
- Concatenation.
- Code Snippet Example 1.
- Code Snippet Example 2.

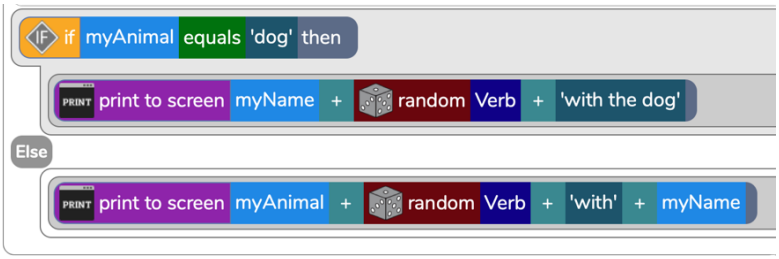
Preparation

- Set Concatenation as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:



- Set Free Code Gorilla as a 2Do.

Activities

Introduction	Display slide 2 and outline the lesson aims. Display slide 3 and outline the success criteria.
Vocabulary	Slide 4 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
	Display slide 5. Use the slide to introduce the word 'Concatenation' and go through the discussion points. Can they think of any coding they have done where they may have used concatenation? They may say they have added a score into a 'Well done' message at the end of a game.
Activity 1: Concatenation	Display slide 6. Tell them you have set a 2Code concatenation task as a 2Do for them in Purple Mash. Ask them to work through it and see how they get on. Before setting them off it you may wish to recap nouns, verbs and adjectives and ask them for some examples of a phrase that includes all three.
Code Snippet Example 1	Display slide 7. Display the Code Snippet Example. But before running the code, can the children suggest what the code will do? Talk through it slowly with the children: There is a myName string set as Archie There is a myAnimal string set as nothing – this is blank – where does the value for this come from? – Ask children to read down and see if they notice that it is set to randomly generate every second. Run it to show how the value of myAnimal is set to a different random animal every second.
Code Snippet Example 2	Display slide 8. Open Code Snippet Example 2 and see if children can read  the code and tell you what will happen if you clicked run now. Run it and notice the errors, see if they can help you debug.

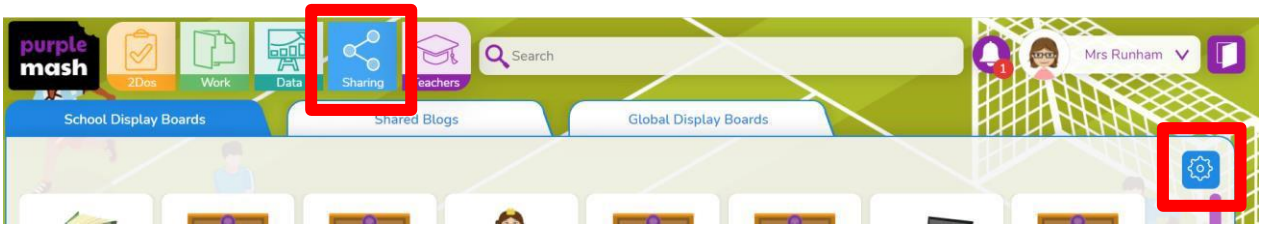
	(Solution) To add the space you need to click on the '+' where you want to add it and then type a space into the box that comes up. Things to note and demonstrate with the children: Use of the Random function – what does random mean? What does the + sign do to text? What would 'Hello + World' produce? (For strings (text) the + will concatenate the pieces of text.) The way 2Code can select a random animal, noun, verb or adjective in order to build sentences. The importance of spaces (the spaces were not correct when you opened it!)
Text Variables	Display slide 9. Start a new 2Code Gorilla document and set a string variable to '1' (the quote marks are required) and + 2 to it. Can children guess what this will make?
Number Variables	Display slide 10. Start a new 2Code Gorilla document and set a number variable to 1 (no quote marks this time) and + 2 to it. Can children guess what this will make?
Explore Text and Number Variables	Display slide 11. Give children some time using Free Code Gorilla to explore text and number variables. What ideas can they come up with to make funny messages?
Vocabulary Overview	Slide 12 can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from slide 3. Children could rate how well they achieved this using a show of hands.

Appendix 1: Display Boards

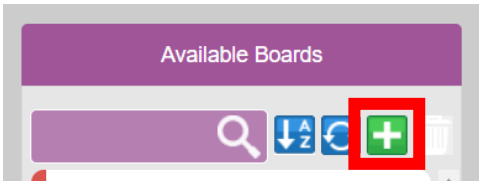
Create the Display Board

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

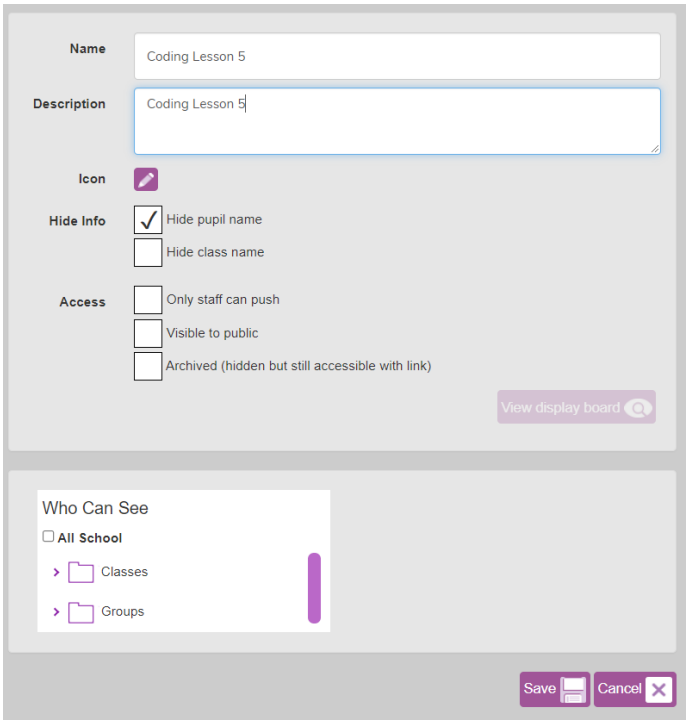
1. Click on the 'Sharing' button to find the Display Board tab, and then click on the settings cog:



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':

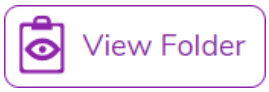


4. Exit Display Board settings:

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:

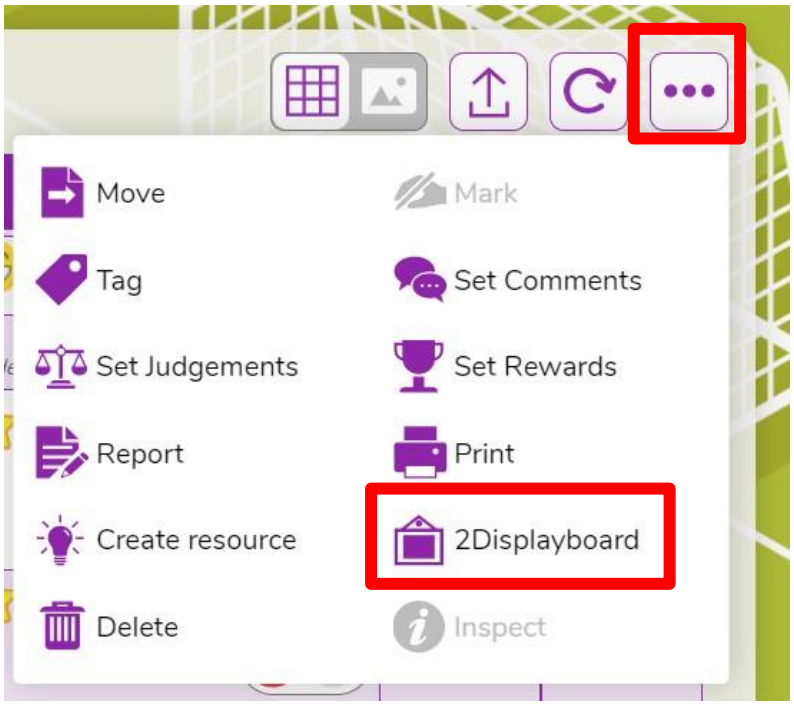
1. Click on 'View Folder' from the 2Do:



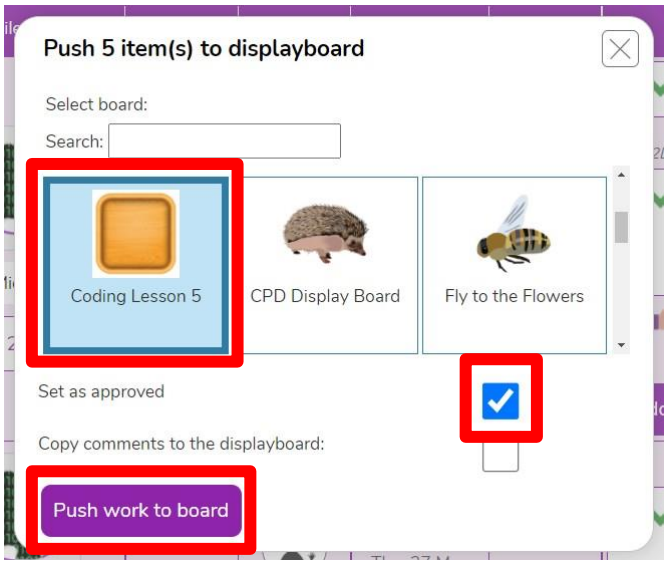
(or navigate to the work you want to share in the Work area).

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':



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

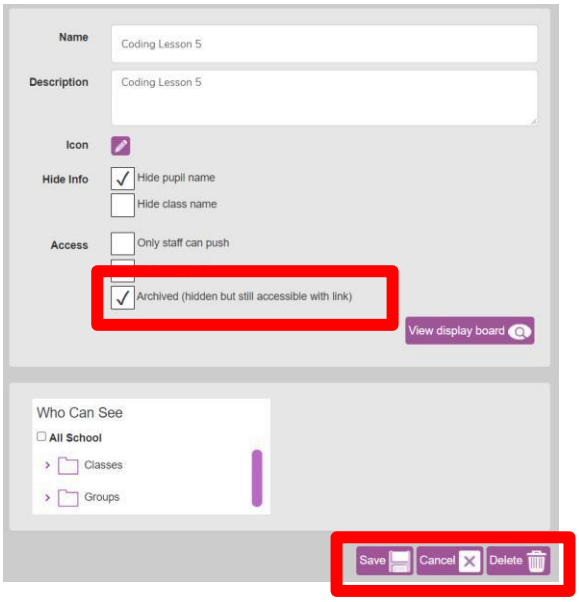


5. Click on 'Sharing' button and then on the display board, you should see the work you've added. It can be deleted by clicking on 'Edit' at the top of the board, then clicking work and then delete. This will remove it from the display board, it won't delete it from Purple Mash.

Deleting or Archiving a Display Board:

When you've finished the lesson, you can return to the Displayboard 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.
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.



Assessment Guidance

The unit overview for Year 5 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	With support, children can begin to create more complex programs that include different types of events in their code (Unit 5.1 Lesson 1). They are beginning to understand what simulations are and with support they have formulated an algorithm for a simple traffic light sequence (Unit 5.1 Lesson 2). As their coding becomes more complex, they will require support to tackle debugging in a logical rather than a trial-and-error method. Children are beginning to understand how decomposition and abstraction are used in computer programming and with support can break a given process down into parts. (Unit 5.1 Lesson 3) They will usually require support to make use of co-ordinates and variables in their code (Unit 5.1 Lesson 4-6).
Expected	Children can create more complex programs and are beginning to understand that there are ways to simplify code to make their programming more efficient. They are able to recall and apply previous coding knowledge in their code. (Unit 5.1 Lessons 1 and 4). Children understand what simulations are and can formulate and program an algorithm for an observed traffic light sequence. (Unit 5.1 Lesson 2). Children understand the processes of decomposition and abstraction and can apply this knowledge when planning algorithms for a program. (Unit 5.1 Lesson 3). Children can include sequence, selection and repetition into code as well as use functions to make their programming more efficient. (Unit 5.1 Lesson 4). Children understand what a physical system is and can consider how they can program objects to behave like the would in 'real life'. Children test and debug their program as they go and can use logical methods to identify the approximate cause of any bugs but might need support to identify the specific line of code that is causing the problem. Children begin to understand how functions work (Unit 5.1 Lesson 4). Children understand that there are different variable types and begin to explore how they can be used (Unit 5.1 Lesson 5).

Assessment Guidance	
	Children can 'read' others' code and predict what will happen in a program which helps them to correct errors. They can also make good attempts to fix their own bugs as their coding becomes more complex (Unit 5.1 Lesson 6). Throughout this unit, children will demonstrate that they are open to feedback from both the teacher and fellow peers on their programs, specifically where they are expected to improve or create a game.
Exceeding	Children can create more complex programs and understand that there are ways to simplify code to make their programming more efficient. With ease, they are able to recall and apply previous coding knowledge in their code (Unit 5.1 Lesson 1). Children can write algorithms for and program simulations, they easily adapt their code to (Unit 5.1 Lesson 2). Children understand the processes of decomposition and abstraction and naturally apply this knowledge when planning algorithms for programs beyond the point at which it was taught (Unit 5.1 Lesson 3). Children intuitively grasp the concepts of selection, repetition and variables. They like to challenge themselves to combine these with other coding structures to personalise and to improve their programs. They understand how to use functions to improve efficiency (Unit 5.2 Lessons 4-5). Children understand and can apply mathematical concepts including co-ordinates, angles and negative numbers with ease when coding (Unit 5.1 Lesson 4). They are also thinking about good structure to their code with a view to debugging such as the use of tabs to organise code and the naming of variables. (Unit 5.1 Lesson 5). Children understand that there are different variable types, can see purpose for them and create and use them with ease when coding. (Unit 5.1 Lesson 5). Children can 'read' others' code and predict what will happen in a program which helps them to correct errors (Unit 5.1 Lesson 6). They are usually successful when attempting to fix their own bugs as their coding becomes more complex.