DigiTech Scheme of Work

Unit 6.1 – Coding 2023

Contents

Introduction

This unit consists of six lessons that assume children have followed the Coding Scheme of Work in Years 1 to 5. 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 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 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.

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/

You can relate this to the way that some teachers follow a progression in Literacy teaching 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.

Year 6 – Medium Term Plan

Lesson	Title	Aims (Objectives)	Success Criteria
1 &2	Designing and Making a more Complex Program	• To design a playable game with a timer and a score. • To plan and use selection and variables. • To understand how the launch command works.	• Children can plan a program which includes a timer and a score. • Children can follow their plans to create a program. • Children can debug when things do not run as expected.
3	Using Functions	• To use functions and understand why they are useful. • To understand how functions are created and called.	• Children can create a program that makes use of functions. • Children can create a program that uses multiple functions with the code arranged in tabs. • Children can explain how their code executes when their program is run.
4	Flowcharts and Control Simulations	• To use flowcharts to test and debug a program. • To create a simulation of a room in which devices can be controlled.	• Children can follow flowcharts to create and debug code. • Children can create flowcharts for procedures. • Children can be creative with the way they code to generate novel visual effects.
5	User Input	• To understand the different options of generating user input in 2Code. • To understand how user input can be used in a program.	• Children can code programs that take text input from the user and use this in the program. • Children can attribute variables to user input. • Children are aware of the need to code for all possibilities when using user input.
6	Using Text-based Adventures	• To understand how 2Code can be used to make a text-based adventure game.	• Children can follow through the code of how a text adventure can be programmed in 2Code. • Children can design their own text-based adventure game based on one they have played. • Children can adapt an existing text adventure so it reflects their own ideas.

Lessons 1 and 2 - Designing and Making a More Complex Program

Aims

- To design a playable game with a timer and a score.
- To plan and use selection and variables.
- To understand how the launch command works.

Success Criteria

- Children can plan a program which includes a timer and a score.
- Children can follow their plans to create a program.
- Children can debug when things do not run as expected.

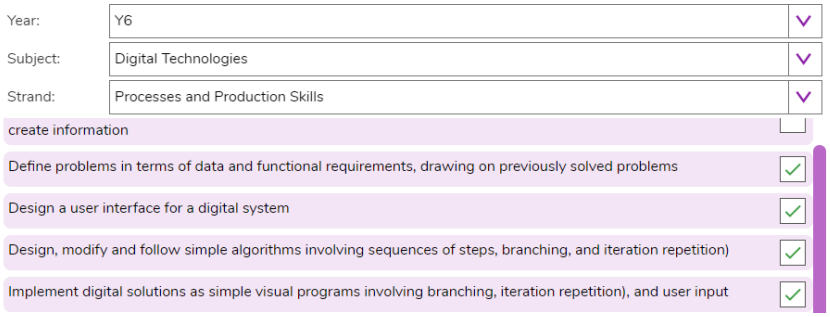
Resources

Unless otherwise stated, all resources can be found on the main unit 6.1 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.

- Coding Vocabulary Quiz Y6
- 2Code Game Planner, Leaflet - Coding Planner,
- Storyboard Template
- Making a Timer and Score Pad guide.
- Splatty Bug.
- 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

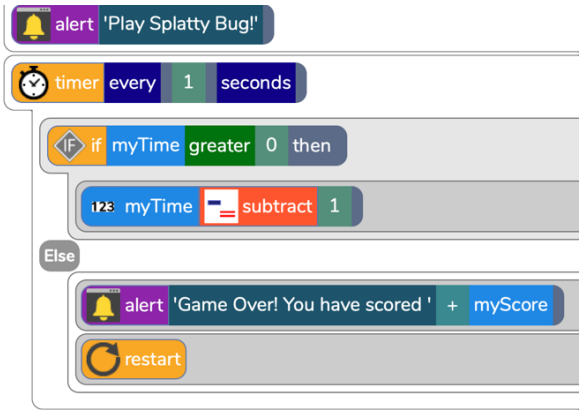
- Set Free Code Gorilla as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:

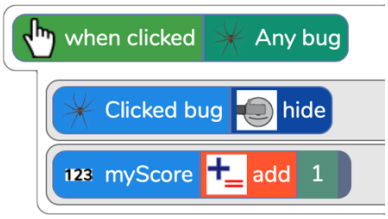
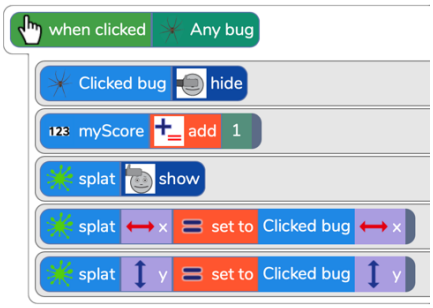


- Print/ copy your choice or a mixture of 2Code Game Planner, Leaflet - Coding Planner, Storyboard template for children to use to plan their programs.
- Open 2 Tabs, Splatty Bug in one and Free Code Gorilla in another.
- Create a display board for the class to share their programs to. Details of how to do this are given in Appendix 1

Lesson 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 Coding Vocabulary Quiz Y6 as a class to review prior learning. 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. Use slide 5 which has definitions. Explain to children that now they are in Year 6, their coding skills are becoming more sophisticated. Emphasise how important the planning stage for their programs is becoming as a result. Explain that over the first two lessons they will plan and make a game that reminds them of lots of the skills they have learnt in primary school.
	Slides 5 and 6 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.
Splatty Bug	Display slide 7. This lesson will focus on the planning but you will start by making a game together to remind children of some of the skills they have learnt. Start the practical part of the lesson by creating a game with them using one of the guided lessons in 2Code. Display Splatty Bug on the board and complete it together as a class. You could complete it on the board or get children to open it as well and complete it with you one step at a time. 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.
Splatty Bug: Stage 1	Display slide 8 to remind them about decomposition and explain that Splatty Bug asks us to add code for one part of the program at a time.

	Stage 1: Start by looking at the design together and ask children to predict how the game will work. Then add the code:  Identify myTime as a variable – in Design View you can see the value of myTime indicates the 'Time left'. Review number variables – what they are and how they work.
Splatty Bug: Stage 2	Display slide 9 and use it to go through stage 2 with the children. Stage 2: Add in a timer that decreases the myTime variable by 1 every second and triggers an alert when the time has run out. Add a restart block at the end.  This section of code will now set the playable time of the game to 10 seconds, alert the player at the end and restart the game. Review selection – IF/ ELSE statements and discuss how they work.

Splatty Bug: Stage 3	Display slide 10. Go through stage 3 Stage 3: Add in a click event that hides the bug that has been clicked on and increases the score. Once you have added 'When clicked Any Bug' you will need to drag across  and then select  to include the 'Clicked bug' variable – the value of which will be set by the click event when it is run:  Discuss simplifying the code and how sometimes variables are used to reduce the amount of code needed.
Splatty Bug: Stage 4	Display slide 11 and go through stage 4. Stage 4: Add code to show the splat and set its X and Y values to equal those of 'Clicked bug'.  Review use of co-ordinates and how they help position objects in a scene.
Splatty Bug: Stage 5	Display slide 12 and use this to go through stage 5. Stage 5: Read the information and click on the 'Info button to see all the code that you would have needed to add if you hadn't used the 'Any bug' feature and 'Clicked bug' variable in stage 3. Take suggestions from the children as to how this game could be further developed. Can anybody suggest why computer programmers are often called developers? Save the final version of the game you've made.

Free Code Gorilla	Display slide 13. Open Free Code Gorilla in a new tab and remind children how to add backgrounds and objects – reminding them that different object types have different options in the attributes table and different possible actions in the code. Ask children to open Free Code Gorilla from their 2Dos area then browse the different backgrounds and objects for a short while to get ideas for the game they will design. Introduce children to the Launch command and demonstrate how it can be used in 2Code to Launch another activity – show how this works for a Purple Mash activity and/ or an external website.
Planning a Program	Display slide 14. The aim for today is to design a game that includes timing and scoring. Can the children suggest any ideas from a game they might have played? Explain to them that you want them to start with a simple version of the game that they can then develop and enhance once it's working. Remind them that this is called a high level of abstraction. Show the children the planning proforma and explain the types of things that should be included in each section, for example, the objects, the events that will occur and any variables required to keep a track of things. They should have Free Code Gorilla in front of them so they can explore background and clipart libraries etc, to help them plan. Use the slide to go through the things they could include. Give children time to think about what they want their games to do and plan in detail before they start coding. At the end of this lesson, let children swap their plans and review each other's.

Lesson 2 Activities

Make Your Computer Game	Display slide 15. Ask children to get out their program plans from last lesson and complete them if they haven't already. Children to open Free Code Gorilla from their 2Dos to create programs using their plans – their games should feature at least a timer and a score. Differentiation: Children can be given copies of the Making a Timer and Score Pad guide to help them implement their designs if needed.
How did you get on?	Ask children to save their program, then use the display board (see Appendix 1) to share great examples with the class, discussing the code that has been used to make them work. Emphasise the importance of the design, code, test and debug process. What challenges did they come across?

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.

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

Lesson 3 - Using Functions

Aims

- To use functions and understand why they are useful.
- To understand how functions are created and called.

Success Criteria

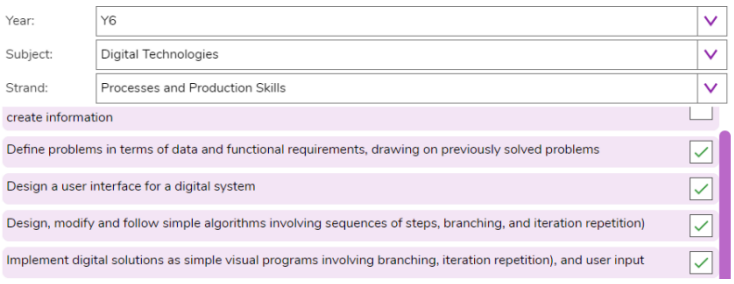
- Children can create a program that makes use of functions.
- Children can create a program that uses multiple functions with the code arranged in tabs.
- Children can explain how their code executes when their program is run.

Resources

- Functions.
- Free Code Gorilla found on the main 2Code screen in the Gorilla section.
- For further guidance on Functions (otherwise known as Procedures) you could use Year 4 Unit 4.5 Using Logo – particularly Lesson 4.

Preparation

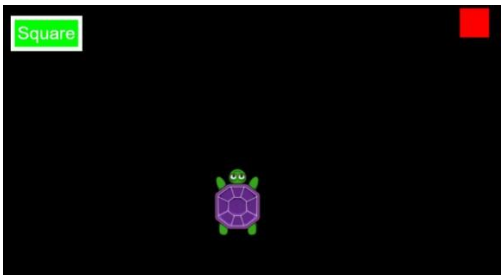
- Set Functions as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:

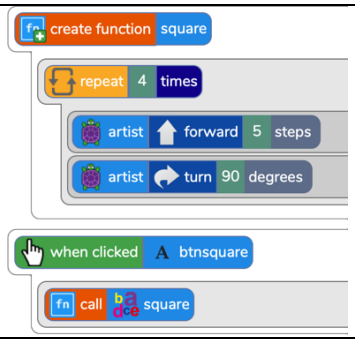


- Set Free Code Gorilla as a 2Do.

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. Explain that in this lesson we will be learning more about functions. Ask children to tell you where they have used functions before – they used them in Year 5 Lesson 4 'Friction and Functions' when they made a Football Game using Freecode. They also practised using them in the Year 5 lesson introducing them to strings.
Activity 1: Functions	Display slide 6. Ask children to log into Purple Mash and have a go at the Functions guided lesson that you've set as a 2Do. Review children's progress and discuss how they've got on using functions.
Free Code Gorilla	Display slide 7. Use this as a guided and shared teaching opportunity before children make their own Turtle Game later in the lesson. Children could open their own version of Free Code Gorilla. Open Free Code Gorilla by clicking on the 'Preview' button in the 2Do. Click on 'Design', then add a turtle object, a shape object, and a text object.
	Display slide 8. Name the turtle object 'artist' in the attributes table that appears to the left when you click on it. Click on the shape and name it 'red'. Set the number of sides to 4, the size to 1 and the angle to 45. Name the text object 'btnsquare', double-click on it and change the text to 'Square'. Show the children how to edit the look of the text object using the attributes table (some of the attributes you could change are show below). Your scene may look something like the this:  Save the work and click on 'See Code.
Make a Function	Display slide 9. Ask the children to help you make a function named Square which will program the turtle object to draw a square. Children will need to guess the number of steps and then debug this code, if necessary, once they have tested it. The function needs to be Called for it to run. Call the function when the btnsquare (the text object) is clicked.

	
Did it Work?	Display slide 10. With the code on the previous slide, the turtle should move in a square shape, but it won't draw a square because you haven't programmed a 'Pen down' command. Ask children to help you look at the code to see if it can be developed in a way that makes the turtle draw the square. Your code might now look like either of the examples shown on the slide. Whichever it looks like, run the program, click on the turtle object and watch it draw a square. Stop running the program and look back at the code, see if the children can spot the alternative place to put the pen down/ pen up commands to achieve the same outcome. Can they help you further develop the code so that when the red square is clicked, it changes the pen colour to red?
Activity 2: Make Your Own Turtle Program	Display slide 11. Ask children to open Free Code Gorilla from their 2Dos area and have a go at creating a turtle program using functions. What other shapes can they create functions for? Show them the images to illustrate the type of program you are expecting them to create. Tell children to arrange their code into tabs so it is easier to debug – remind them about decomposition.
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. Review children work together against the lesson aims and success criteria – this could be done by sharing some good examples from the 2Dos folder. Can children think of another program they use that works a bit like this? They may recognise it is being a bit like Logo and functions as working a bit like procedures in Logo.

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

Lesson 4 – Flowcharts and Control Simulations

Aims

- To use flowcharts to test and debug a program.
- To create a simulation of a room in which devices can be controlled.

Success criteria

- Children can follow flowcharts to create and debug code.
- Children can create flowcharts for procedures.
- Children can be creative with the way they code to generate novel visual effects.

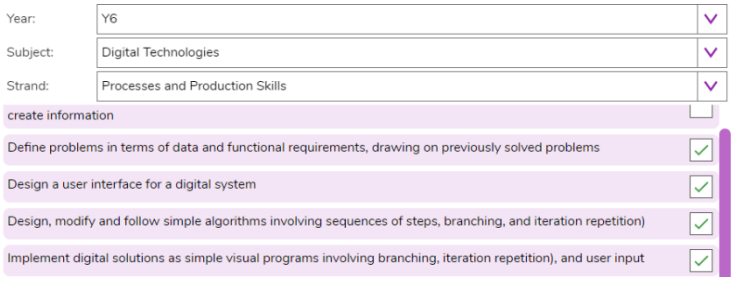
Resources

Unless otherwise stated, all resources can be found on the main unit 6.1 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.

- Billy's Bedroom Simulation.
- Billy's Bedroom Flowchart.
- A piece of plain paper and a pencil for each child.

Preparation

- Set Billy's Bedroom Simulation as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:



- Print/ copy Billy's Bedroom Flowchart enough for one each for the children.

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 explain that in this lesson we are going to work on a simulation, ask children if they can remember what a simulation is – they covered it in Year 5. Reveal definition on slide.
Billy's Bedroom Flowcharts	Display slide 6. Open the Billy's Bedroom Flowchart – note that there are several flowcharts on one page here, and each flowchart represents a procedure for something that happens in Billy's bedroom. Give each child a printed copy of Billy's Bedroom Flowchart. Give children 6 minutes to draw a bedroom that could be Billy's – emphasise that it doesn't need to be a work of art, it's just a quick sketch. What is in the room?
Billy's Bedroom Simulation	Display slide 7. Open Billy's Bedroom Simulation in 2Code and share with the children. Does it contain all the same things as your picture? Probably, but quite likely in different places. Click on the Run button to run the code. Watch it for a while, and notice it switching between day and night. Ask children: What are the differences? What might happen at night that doesn't happen during the day? How long is the day/ night? Give children a couple of minutes in talking partners to look at their printed flowcharts and predict what will happen when different objects are clicked on.
	Display slide 8. Run the simulation. Ask children to suggest objects for you to click on, does the simulation run like they were expecting? After you click on each discuss if they think it has been coded correctly, or if they think it needs correcting/ debugging.
Activity 1: Complete Billy's Bedroom Simulation	Display slide 9. Set the children the task of interpreting the flowcharts, debugging existing code and getting the rest of the devices in Billy's room working.

	- There are no flowcharts for some objects (car, robot, computer) they can plan the procedures for these in flowcharts on paper or in 2Chart in another tab. - They might need to add more objects e.g. there is no remote control for the car, they will need to add buttons to look like a remote control. They can look at the buttons on the drawer knobs to get an idea for this. - There are functions to make the fairies fly, but these do not have any code in them. They should start with the things that look simplest and then try the harder ones. They could add more devices and create flowcharts for them if they have time.
Vocabulary Overview	Slide 10 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.

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

Lesson 5 – User Input

Aims

- To understand the different options of generating user input in 2Code.
- To understand how user input can be used in a program.

Success criteria

- Children can code programs that take text input from the user and use this in the program.
- Children can attribute variables to user input.
- Children are aware of the need to code for all possibilities when using user input.

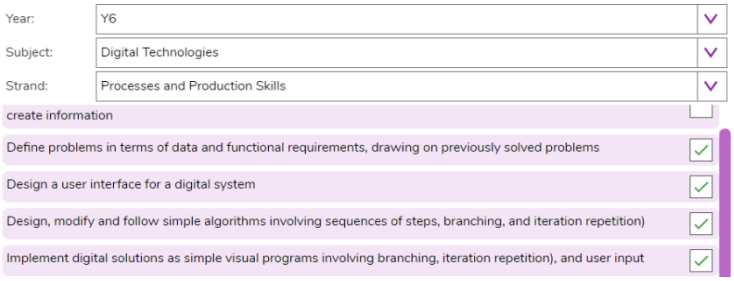
Resources

Unless otherwise stated, all resources can be found on the main unit 6.1 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 cards.
- Aliens Database.
- Guess the Alien.
- Is It Raining and Reginald Rocket Examples from Y4.
- User Input Example 1.
- User Input Example 2.
- Free Code Gorilla found on the main 2Code screen in the Gorilla section.

Preparation

- Set Guess the Alien as a 2Do. Add in Aliens Database and Alien Cards as supporting documents – you will need to save these documents to your work folder before you set the 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:



- Print/ copy Alien cards enough for one each for the children.

- Open 4 tabs: User Input Example 1, User Input Example 2, Is It Raining and Reginald Rocket.
- 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. Reveal the content of the slide and go through 'User Input'. Open Reginald Rocket and Is it Raining in separate tabs. Look at the code, predict and then run both.
Example Code	Display slide 6. Reveal the slide's contents to look at the example code and discuss concatenation. Open User Input Example Code 1 and with the children, add the missing full stop. Run the code.
	Display slide 7. Show children User Input Example 2. Discuss setting the input to a variable (number or string) and then manipulating it. What happens if you give a word answer for this last example?
Activity 1: Using Input	Display slide 8. Ask the children to Start Free Code Gorilla and explore using the get input command and the prompt for input command. Challenge them to have a go at adding code so that the program uses the input in a string when it is collected.
Alien Game	Use slide 9. Explain to children that in this lesson they will be making an interactive game using input from the user. Display the Alien cards or slide 8 on the board and find out about the aliens. Ask the children to come up with questions to identify them, you could ask them to do this with talking partners.
Guess the Alien	Display slide 10. Open Guess the Alien Game on the board and read the code in the first tab – the code for the 'Introduction'. Ask children to explain how they think this game starts – they could discuss it and then feedback. Click on Run to run the game, choose the alien called Zinky (at the moment Zinky will work but none of the others have been

	programmed yet). Work through the questions until the program guesses that the alien is Zinky.
	Display slide 11. Look at the code in the second tab – the code for 'Red Aliens' and discuss how it works with the children. What do they think the next steps to completing the program are? Add code for the red alien with 2 eyes. Add code for the other colours of aliens; after colour, the questions do not have to be about number of eyes, but they do have to be about appearance.
Activity 2: Guess the Alien	Display slide 12. Ask the children to open Guess the Alien from their 2Dos and see if they can add more to it. Give them a copy of the Alien Cards or have the pictures of the Aliens on the board so they have a point of reference. Remind them that they can use the Aliens Database to find out more detail about the aliens. Remind them about decomposition and using tabs.
How did you get on?	Display slide 13. Share children's work 2Displayboard (see Appendix 1) and play a couple of the games together to share children's programs and celebrate achievements.
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.

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

Lesson 6 – Text-Based Adventures

Note: This lesson covers quite a few aspects of text adventures; the adventure is revisited in unit 6.5. You might choose to focus on this session purely on mapping the game and exploring a text adventure or you might wish to extend this session over two lessons to look at the code in more detail.

Aim

- To understand how 2Code can be used to make a text-based adventure game.

Success Criteria

- Children can follow through the code of how a text adventure can be programmed in 2Code.
- Children can design their own text-based adventure game based on one they have played.
- Children can adapt an existing text adventure so it reflects their own ideas.

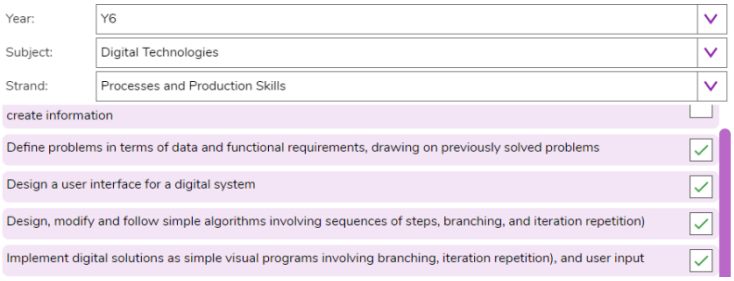
Resources

Unless otherwise stated, all resources can be found on the main unit 6.1 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.

- Y6 Text Adventure.
- Pencils and paper for children to sketch maps on.
- Text Adventure Functions.

Preparation

- Set Y6 Text Adventure as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier:



- Print/ copy Text Adventure Functions to hand out as a supporting document.
- 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 aim. 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 'Text Adventure'. Explain to children that they are going to begin by playing a simple text adventure written using 2Code, and then they will have a chance to change the code to make the adventure their own.
Activity 1: Year 6 Text Adventure	Display slide 6. Ask children to open the Y6 Text Adventure from their 2Dos and press Run to get started. Give the children some time to play the adventure, see who is first to solve it. Ask them to sketch a map as they are playing it. See if they can work out that the aim of the game is to collect the diamond and give it to the wizard, who will then provide them with a key to escape.
Using a Map	Display slide 7. Once children have had time to play, compare the maps that they have drawn and discuss how having a map is helpful for solving the game. It is useful for everyone to have a simple map when discussing the code. If children's maps are varied, let them refer to the image on the slide. The next stage is to look at the code. Children will be making changes to the code, so it is worth talking about the need to save their files frequently after testing that they work. It can be easy to remove some vital code and then not know how to get the code back to how it was before so by saving correct code after each successful change you can always 'exit without saving' and then click on 'continue' in the 2Do to open your last saved file. Remind children of the undo button which appears when you have changed something so that you can undo; often this will fix the problem.
Number the Rooms	Display slide 8. In the code, each room has been given a number. Players start in room 0. Look through the code together and try to write the numbers on the map. Look at your map, Slide 8 has help for rooms 0, 1 and 2. Now that the rooms are labelled with the numbers, look through the code in even more detail – starting by looking at the variables, then by looking at functions, then by looking at how the repeat command is used followed by how

	the game concludes. This will help children understand how the game works and check the numbers on their rooms are correct.
Variables	Display slide 9. Use the slide to explain to the children that the first 5 lines of the code are where various variables are set. Articulate the following points to children: • Do the names of the variables provide clues as to what they are for? They should do and it's the reason that you should always try to name variables well. • Once your coding gets more complex it is very confusing if the variables are all called things like 'MyNumber1'.
Functions	Display slide 10. Use the slide to explore the next section of code – Functions. Articulate the following points to children: • The functions are only run when they get called by the main code. • Each function starts with a prompt for input – because when you get into each room you are given some information and asked a question, and what happens next depends on your answer. • By looking at the prompt we can confirm that room 1 is the mirror room – you could use the information in the functions to check the room numbers on your map. *In the resources, you'll find Text Adventure Functions Code. Some children might find it helpful to have printed versions that they can refer to when changing code.
Repeat Until	Display slide 11. Use the slide to explore the next section of code – Repeat. Articulate the following points to children: • This code is after the functions but will run when the program is started. • At the start of the code the variable 'room' was set to equal 0, so when the program is run and the first IF statement is checked, it will be true and the first function to be called will be 'roomIs0'. • When the player moves rooms the 'room' variable will change. This repeat until command will run through the IF/ ELSE statements until it finds a statement that is true and then it will call the function for that room. • There is an IF/ ELSE statement for each room.
Conclusion	Display slide 12 and explore the code that runs when the 'finished' variable =1. Can the children explain what this does?
Activity 2: Your Own Text Adventure	Display slide 13. Ask children to draw a blank plan of the building and plan their own room layouts and features. Leave the doors in the same place. Challenge them to plan to change the code for the text adventure they have been playing and exploring and make their own text adventure: Reveal ideas on the slide.

	Display slide 14. Challenge children to change the code for the text adventure they have been playing and exploring and make their own text adventure. NOTE: Sometimes when text is changed it reverts the text to what it was before when they go to change another part of the code. When children change text, tell them to make sure they click out and check it has changed before going on to change something else. If they are having problems, they could drag in a new prompt box enter their text into that and delete the original one.
How did you get on?	Display slide 15. Children's work could be shared to a display board for each of them to try out.
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.

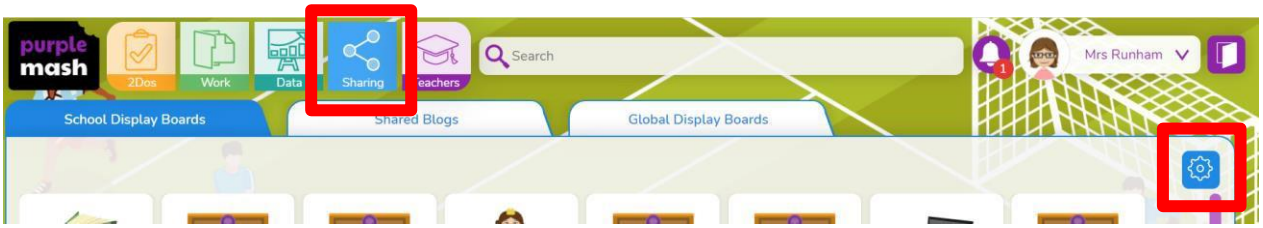
Remember to close your 2Dos when you have finished the lesson.

Appendix 1: Display Boards

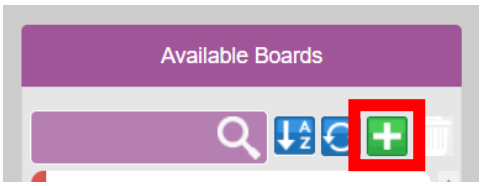
Create the Display Board

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

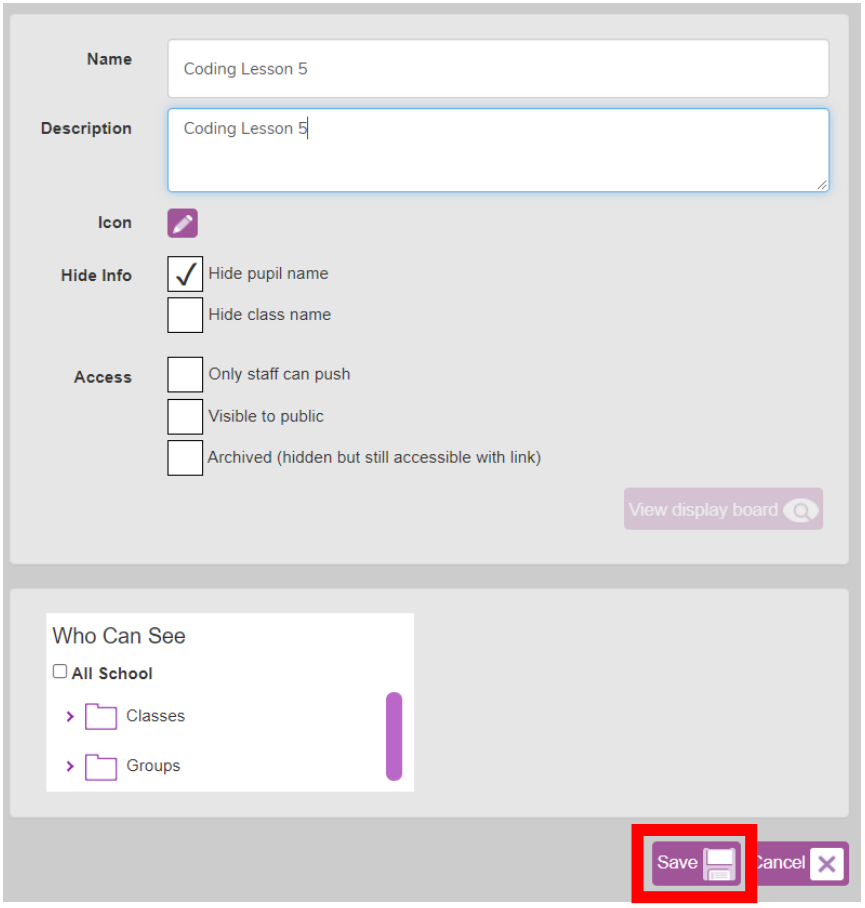
1. Click on the 'Sharing' button to find the Display Board tab, and then click on the settings cog:



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':

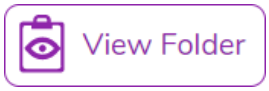


4. Exit Display Board settings:

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:

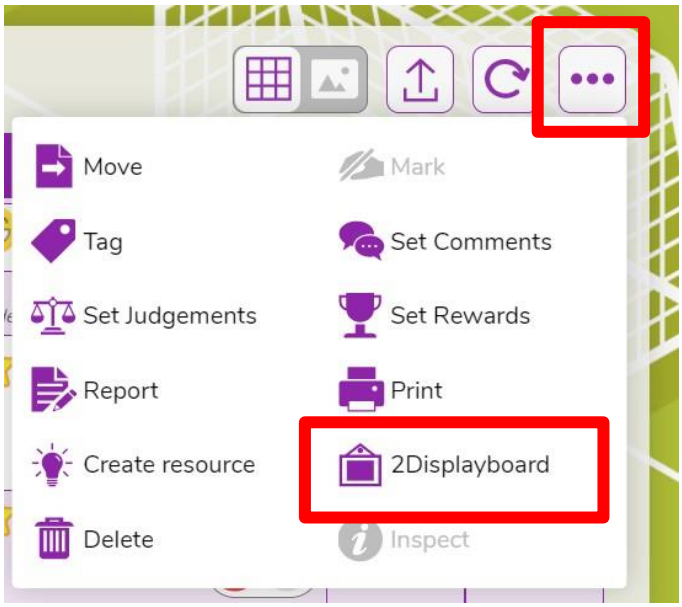
1. Click on 'View Folder' from the 2Do:



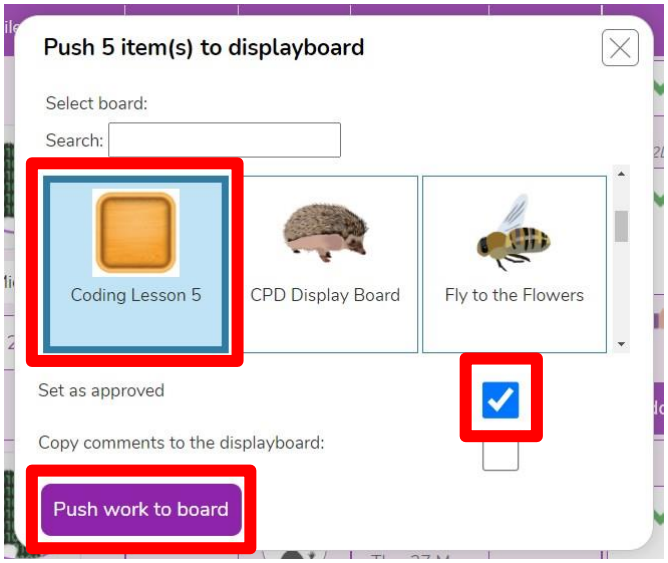
(or navigate to the work you want to share in the Work area).

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':

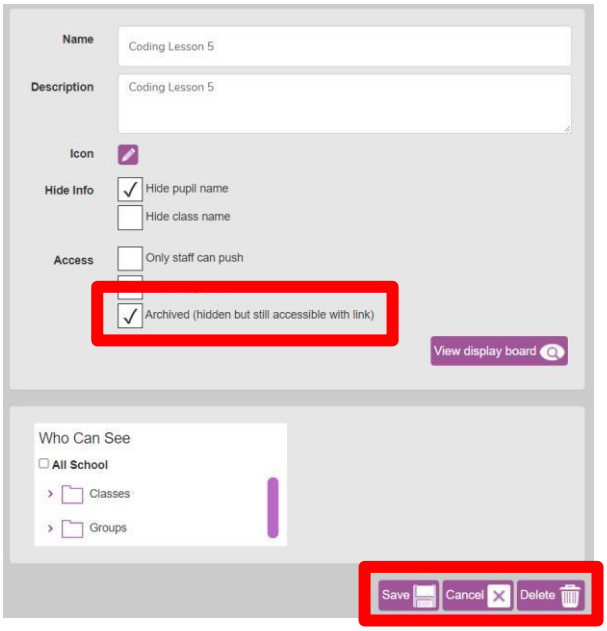


5. Click on 'Sharing' button and then on the display board, you should see the work you've added. It can be deleted by clicking on 'Edit' at the top of the board, then clicking work and then delete. This will remove it from the display board, it won't delete it from Purple Mash.

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 6 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	Children are beginning to be able to turn a more complex programming task into an algorithm by identifying the important aspects of the task (abstraction) and then decomposing them in a logical way with support (Unit 6.1 Lessons 1 and 2). They can then use this design to write a program using 2Code. Children understand sequence, selection and repetition in programs and can use them in their simplest forms. They will require support when combing these aspects e.g. using selection within a repeat in a game (Unit 6.1 Lessons 1, 2 and 6). With support, children can plan, design and create a simple program that includes a single variable relating to timing. They can also include a button which will launch another program (Unit 6.1 Lessons 1 and 2). They will usually require support to make use of variables and manipulate variables in their code and in understanding the way that functions are beneficial (Unit 6.1 Lessons 1-4). As their coding becomes more complex, they will require support to tackle debugging in a logical rather than a trial-and-error method. Children can make good attempts to 'read' code and predict what will happen in a program (Unit 6.1 Lessons 4-6). They can usually interpret a program in parts but will need support to put the separate parts of a complex algorithm or program together to explain the program as a whole (Unit 6.1 Lesson 6).
Expected	Children are beginning to be able to turn a more complex programming task into an algorithm by identifying the important aspects of the task (abstraction) and then decomposing them in a logical way using their knowledge of possible coding structures and applying skills from previous programs. They can then use this design to write a program using 2Code (Unit 6.1 Lessons 1 and 2). Children can translate algorithms that include sequence, selection and repetition into code and their own designs show that they are thinking of how to accomplish the set task in code utilising such structures including nesting structures within each other (Unit 6.1 Lessons 1-6). Children can plan, design and create a program that includes variables relating to timing and scoring along with buttons which launch other programs (Unit 6.1 Lessons 1 and 2). Furthermore, children will consider how to organise their code using multiple tabs (Unit 6.1 Lessons 1, 2, 3 and 5). They use functions within their code to eradicate unnecessary code and make their programming more efficient (Unit 6.1 Lesson 3).

Assessment Guidance	
	Their coding displays an understanding of the function of variables in coding (Unit 6.1 Lessons 1 and 2 and Lesson 6), outputs such as sound and movement (Unit 6.1 Lessons 1 and 2), inputs from the user of the program such as button clicks (Unit 6.1 Lessons 3, 4 & 5) and the value of Functions (Unit 6.1 Lesson 3). Children can make good attempts to 'read' code and predict what will happen in a program (Unit 6.1 Lessons 4 and 6). They can usually interpret a program in parts and can make logical attempts to put the separate parts of a complex algorithm or program together to explain the program as a whole (Unit 6.1 Lesson 6). 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 as the complexity of the programs increases. They try to improve and debug their own programs (Unit 6.1 All Lessons). Within their programs, they can use features such as interactivity with the end users with the desired effect of adding greater impact. (Unit 6.1. Lesson 5 and 6). Most children demonstrate a secure understanding of the impact of changing the position of instructions within 2Code. With this knowledge, they can demonstrate use of the tabs feature to carefully section code for the intention of easier debugging and less code error, as their coding becomes more complex.
Exceeding	Children can turn a more complex programming task into an algorithm by identifying the important aspects of the task (abstraction) and then decomposing them in a logical way using their knowledge of possible coding structures and applying skills from previous programs. They can then use this design to write a program using 2Code (Unit 6.1 Lessons 1 and 2). Children's designs show that they are thinking both of the required task, and of how to accomplish this in code. Children test and debug their program as they go and can use logical methods to identify the approximate cause of any bugs then test systematically to identify the specific line of code that is causing the problem. Children intuitively grasp the concepts of selection, repetition and variables and make use of the various commands to use input from users and produce output including sound and movement. Children like to challenge themselves to combine these with other coding structures to achieve the effects that they design to personalise and to improve their programs (Unit 6.1 Lessons 4-6). They are also thinking about good structure to their code with a view to debugging such as the use of tabs and functions to organise code and the naming of variables.