purple mash

CRASH COURSE

Computing Scheme of Work

Year 2 Coding Crash course

For pupils in Year 2 who have not used 2Code previously.

Contents

Introduction

Differentiation

Children will need to be able to drag and drop to move code blocks around. If pupils have not had much practice with this then there are several example activities within the Activities section of 2DIY that help pupils to practise these skills in preparation: 2DIY activities to practise drag and drop. Within each category of activity, look for the example file then press the Play button. If the pupils have not used Purple Mash before, spend some time showing them how to log in and how to get to 2Code.
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'.
The crash-course aims to prepare pupils for using the Computing Scheme of Work Coding unit in year 3. To enhance pupils' ability to code and understand the process of coding and design, children should have had as many of the following experiences as possible:

Challenges

When using the guided activities, pupils should have attempted the challenges at the end of the guided lessons in 2Code and come up with solutions to these either individually or using shared coding as a group or class.

Free coding

Pupils will benefit from spending some time using:

- Y1-2 Free Code Chimp (or Free Code Scenes)
- Y3-4 Free Code Gibbon
- Y5-6 Free Code Gorilla

To create their own programs.
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 build up their vocabulary of coding words. The Chimp activities provide further practice of the concepts that the children will be learning and can be used as extension activities. More able children should be encouraged to explore the options in 2Code and use them to develop their program, such as by changing object images and altering their scale in 2Code.
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 undertaking in the 'real world'. More able children can be encouraged to support their peers, helping them to understand but without doing the work for them.

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

PRIMM

The coding lessons in these units are structured around the **PRIMM** approach. The whole approach may take place during a lesson or series of lessons.
Predict… what this code will do
Run… the code to check your prediction
Investigate… trace through the code to see if you were correct
Modify… the code to add detail, change actions/outcome
Make… a new program that uses the same ideas in a different way. Get creative!
Often lessons will start by looking at existing code, asking the children to 'read' it and make **P**redictions to what they think will happen when the code is run. You'll then **R**un the code and give them time to discuss what happens and relate it back to their predictions. You'll spend time with them **I**nvestigating 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 **M**odify 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 **M**ake their own program from scratch.

Levels of Scaffolded coding tasks

You can support pupils' learning and understanding by using different degrees of scaffolding when teaching pupils to code. The lessons provide many of these levels of scaffolding within them and using Free Code Chimp, Gibbon and Gorilla enables pupils to clarify their thinking and practise their skills. These are not progressive levels; pupils 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	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.
Least scaffolded	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 pupils 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	Success Criteria
1	Introduction to Coding: Objects and Actions	• Children can explain what coding is. • Children know that for the computer to make something happen, it needs to follow clear instructions. • Children can create a program using **code** blocks. • Children can use **object** and **action code** blocks.
2	Events	• Children can create a simple program using **code** blocks. • Children can use **event**, **object** and **action** code blocks.
3	Algorithms	• Children can explain that an **algorithm** is a set of instructions. • Children can describe the **algorithms** they created. • Children can explain that for the computer to make something happen, it needs to follow clear instructions.
4	Collision Detection	• Children can plan an **algorithm** that includes **collision detection**. • Children can create a program using **collision detection**. • Children read blocks of code and **predict** what will happen when it is **run**.
5	Using a Timer	• Children can create a program that uses a **timer-after command**. • Children can explain what the **timer-after command** does in their program. • Children can **predict** what will happen in a program that includes a **timer-after command**.
6	Different Object Types and Buttons	• Children can create a computer program that includes different **objects** types. • Children can modify the **properties** of an **object**. • Children can use different **events** in their program to make **objects** move. • Children can create a computer program that includes a **button** object. • Children can explain what a **button** does in their program. • Children can modify the properties of a button to fit their program design.

Lesson 1 - Introduction to Coding: Objects and Actions

Aims

- To understand what **coding** means in computing.
- To use code to make a computer program.
- To understand what **objects** and **actions** are.

Success criteria

- Pupils can explain what **coding** is.
- Pupils know that for the computer to make something happen, it needs to follow clear instructions.
- Children can create a program using **code** blocks.
- Children can use **object** and **action code** blocks.

Resources

Unless otherwise stated, all resources can be found on the unit main page. From here, they can be set as 2Dos by clicking on the icon. To preview resources linked to here, right-click and '**open in new tab'** so you do not navigate away from this page.

- Code block cards.
- Fun with Fish Activity. This is on the main 2Code page in the Chimp section.
- Optional: Exercise books to be used as 2Code workbooks for recording coding exercises and designs.

Preparation

- Set Fun with Fish as a 2Do.

Activities

Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.
Introducing Programming	Use **slide 4** discuss the questions and click to reveal further points for discussion.

Activity 1: Teacher is the Programmer	Use **slide 5** to explain to the children that you are now going to be a programmer and they are all going to be robots. Reveal the instructions on the board as symbols. Get children to 'act' out/ follow the instructions you have displayed as symbols – a twirl, a hand next to a toe and a hand next to an ear - the children should twirl, touch their toes then touch their ears.
Activity 2: Using Symbols	Use **slide 6** to display a hand next to an up arrow and see if the children can see that this would be 'hand up'! Ask children to use small whiteboards draw symbols for 'hand down', what about touch nose?
Computers Follow Instructions	Display **slide 7**. Now that children have practised receiving instructions in code represented as symbols, reiterate the introduction using this slide. Explain that a coder writes instructions in code for the computer to follow, this is called the input. These instructions make our programs work, our programs are the output.
Fish in the Sea	Display **slide 8** Fish in the Sea and discuss what you can see – 3 fish in the sea. Ask children what they think those fish could be programmed to do.
Algorithms	Use **slide 9** to explain the word algorithm: the algorithm is not the code but more like a detailed plan for the program.
Code Blocks	Show the children the code on **slide 10**. Explain that these are examples of **code** used to program a computer. Can they suggest ways to combine the cards to make **instructions**? Pair up the children and give the pairs some printed out Code Block Cards. Challenge them to join two blocks to give one clear **instruction** (i.e. tuna - left). Child one should lay out a line of code by joining blocks, then Child two should 'read' the line of code and explain what the **code** would do. Repeat, swapping roles.
Objects and, Actions	Display **slide 11**. Explain to the children that the light blue code blocks represent objects, and the dark blue code blocks represent actions.
Command	Display **slide 12** to introduce the term **Command**. Explain that a single instruction is called a **command**.
Demonstrating: Fun with Fish	Display **slide 13**. Use the icon\link to open the activity and demonstrate (children will eventually open from their 2Dos). Display **slides 13-15**. Use the icon\link to open the activity and demonstrate using the points on the slides. Emphasise the need to give the computer clear instructions for moving the fish. Show the children what to do if they click on the wrong direction - click on the direction again and select the correct one. Show the children where the Play button is to **run** the code and emphasise that the code has programmed the **object** to do an **action.** Show them how to move to the next stage of the activity or stop the code running to make changes. Complete stage 2 together as a class.

Activity: Fun with Fish	Display **slide 16**. Ask children to log in to Purple Mash, go to their 2Dos and click on 'Start' on the Fun with Fish 2Do. Challenge them to complete stages 1 and 2. Ask them to use the **code** blocks to make their Tuna move, and then move onto the next challenge to make the Crab move.
	Use **slide 17** to explain the next stage. Children should complete stage 3 independently. Review progress together - did they get lots of code monkey stars? The maximum is 5; they lose stars for using hints.
	Display **slide 18**. and look at stage 4 together. Take a few suggestions from the class about how to improve the fish tank by adding new **objects** – fish/ crabs – add one new **object** then switch to the code screen to notice it then appears as a blue object code piece, show how to program it to move and test it out using the play button. Add one or two more **objects** and show children how to use the **event**. Ask children to complete the challenge stage and then save their work before they save and exit. Review children's work together against the lesson aims – this could be done by sharing some good examples from the 2Dos folder. Did any children try using the '**when clicked event**'? What did that do?
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.

Lesson 2 - Events

Aim

- To understand what an **event** is.
- To use an **event** to control an **object**.

Success Criteria

- Children can create a simple program using **code** blocks.
- Children can use **event**, **object** and **action** code blocks.

Resources

Unless otherwise stated, all resources can be found on the unit main page. From here, they can be set as 2Dos by clicking on the icon. To preview resources linked to here, right-click and '**open in new tab'** so you do not navigate away from this page.

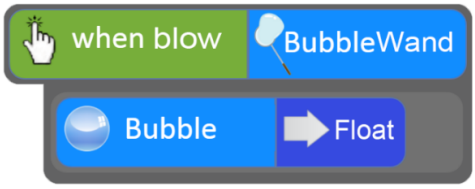
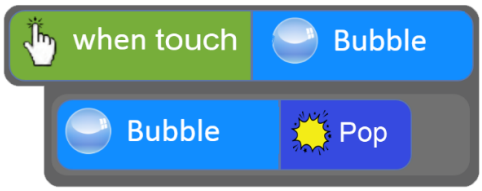
- A pot of bubbles and a bubble wand (usually part of the lid!)
- Bubble Coding
- Example Code

Preparation

- Set Bubble Coding as a 2Do.

Activities

Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.
Events	Display **slide 4**. Start by telling children that in this lesson they will all use **events** in their coding. See if any of them can remember and describe how an event worked in the last stage of Fun with Fish (you could bring up an example of a child's previous work to show them).
Activity 1: Bubbles	Use **slide 5.** Blow bubbles. Ask children: What is the **event**? (What do you do to make something happen?) (blow bubble wand – if they just say 'blow', blow the air nowhere near the bubble wand and ask them if that works! Talk about the need for **precise instructions**). • What are the **objects**? (bubble wand, bubble) • What is the **action**? (float)

	Use the Example Code or display **slide 6.** Show the children what the code could look like for blowing bubbles. 
	Use **slide 7** to talk about the **event**, **object**, **action** (touch – bubble – pop). Ask children to rearrange the following **code** for that (this could be done physically with Example code printed on paper or as a drag into place activity on the board using the code on slide 7 with the slideshow in edit rather then run mode). 
Activity 2: Other Events	Use **slide 8** to discuss other **event** – **object** – **action** examples children might be familiar with (e.g. push – swing – swing - forward, kick – football - football – roll). Rearrange the **code** for the football example (this could be done physically using Example Code printed on paper or as a drag into place activity on the board using slide 8). 
Activity 3: Bubbles Coding	Use **slide 9** to demonstrate click **events** activity before they have a go (slide 10). Open Purple Mash and go to 2Dos, click on 'Preview' within the Bubble Coding 2Do to show children the Bubbles lesson. Look at the available code blocks available and see if children can tell you what they might see when you click to 'Design'. Referring to **event**, **object**, **action**, add code that makes a bubble move up when it is clicked on, add another bubble and make it pop when it is clicked on. Remind children of the play button to **run** the code. Test the **code** you have just added in together, discuss what other **code** they could add to make the other bubbles move.
	Use **slide 10** to set children off independently on Bubbles Coding from 2Dos.
	Use **slide 11** to review progress together as a class – using terminology **event**, **object**, **action**. Have any of the children noticed or tried to use the **sound** button? Demonstrate how it might be used so the bubble makes a **sound** when it pops. Challenge children to improve their program by adding **sounds** and then saving before they exit.

Review Success Criteria	Display **slide 12**. Review the success criteria from slide 3. Children could rate how well they achieved this using a show of hands.

Lesson 3 - Algorithms

Aims

- To understand what an **algorithm** is.
- To create a computer program that follows an **algorithm**.

Success Criteria

- Children can explain that an **algorithm** is a set of instructions.
- Children can describe the **algorithms** they have created.
- Children can explain that for the computer to make something happen, it needs to follow clear instructions.

Resources

Unless otherwise stated, all resources can be found on the main unit 2.1 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.

- Coding Vocabulary Quiz (Year 2 version)
- Two identical sets of any construction toy
- Air Traffic Control.
- (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 Air Traffic Control as a 2Do for your class.
- Build two models using two identical sets of any construction toy - one that follows the instructions and one that does not. An example would be using Lego Duplo to build a bird, one that follows these instructions and one that uses the same bricks but not the instructions. Download the instructions for your model so that you can display them on the board.
- (Optional) Print storyboard templates for program design.

Activities

Introduction	Display **slide 2** and outline the lesson aims.

	Display **slide 3** and outline the success criteria.
Vocabulary	Use the Coding Vocabulary Quiz on **slide 4** as a class to help review the coding vocabulary learnt in Lesson 1. It is set up so that you attempt all questions and then click the hand in button to check the answers. Run through the answers to the questions together. You could use the vocabulary cards to find the answers and display in the classroom.
Algorithms	Use **slides 5 and 6** to introduce today's lesson. Read and discuss the definition of an algorithm.
Activity 1: Lego Models	Display **slide 7**. Show the children the two models you built using identical construction toys - without displaying the instructions. Ask them which is correct? The answer you are looking for is that they are both correct; there is no such thing as 'correct' or 'incorrect' when building creatively. They might prefer one over the other, but both are correct.
	Display **slide 8**. Display the instructions on how to build the model and ask the questions on the slide. If you have enough building materials for the class, they could attempt to follow the **algorithm** to create the model themselves.
Making a Computer Program	With **slide 9**, discuss the process of making a computer program. Look at the design and algorithm for the airport program. This time you want children to concentrate on implementing this **algorithm**. Discuss what the **objects** in this program are (the planes and the helicopter); what are the **actions**? (the planes go up and right); what **events** are used to make these actions happen? (a click **event** is needed to make the objects move).
Activity 2: Air Traffic Control	Display **slide 10.** Demonstrate and remind children that there are hints if they need them, and that once they've clicked on OK they can get back to the hint by clicking on the instruction at the top. Ask children to come up to the front of the class and use the code blocks to make the plane take off when it is clicked on:  Explain that if you click on the plane before the **code executes** it won't take off, they need to make sure the code **executes** first. Get children to watch the code and see what happens when you click on the planes, which bit of code **executes** when?
	Use **slide 11** to set children off on completing stages 1 and 2 of Air Traffic Control themselves.
	Use **slide 12** to support looking at stage 3 together as a class. Discuss what **debugging** is – detecting and fixing any errors in the code. Demonstrate how you can drag code around to move it into a different place and click on actions to change them. Fix the code together and click on Next Challenge. Set children off to complete the **debugging** stage.
	Display **slide 13**. Look at the slide together and revise the main elements of the code view.

Air Traffic Control Final Stage	Display **slide 14**. Talk about the design of Air Traffic Control Final Stage together and ask children to point out the **background** and **objects** in the design. Refer back to the program on **slide 9**. Compare it to the design for our Airport Program..
Algorithm -› Code	Follow **slide 15**; use it to open the final stage of Air Traffic Control in 2Code and click on 'Exit Design' and ask children to help you add in code for the first step of the algorithm (click event to enable a plane to take off). Click on the play button and run the program. Watch what happens when the code is running: why does some of the code go orange? It is when that bit of code executes.
Activity 2: Air Traffic Control	Display **slide 16**. Ask children to return to 2Code and continue working through Air Traffic Control until they reach the final stage, then challenge them to start adding the code to make the first 3 steps of the **algorithm** work.
Saving Your Code	Use **slide 17** to remind children how to save their work and discuss why it is important to save their coding regularly so that they have a working version to go back to.
Air Traffic Control	When most of the class have programmed the **click events**, draw their attention back to **slide 18** and work together to program the last step of the **algorithm** - to make a **sound** play if the helicopter crashes into the yellow plane. When you have talked through the questions on the slide, as an extension ask: could we make a different sound play if the helicopter collides with the purple plane? (You would need two **collision detection** events – one for each plane)
Extension: Air Traffic Control	Use **slide 19** to challenge children to see if they can add **code** for the **collision detection** and complete making the program work. For the optional final step of the **algorithm** children will need to add 2 **actions** to their **collision detection** event. Children can now get creative by adding other planes and runways onto their design and program them. Note: Although you can add more landing strips into the design, you would not normally program them to do actions!
How Did You Get On?	Display **slide 20**. Ask children to work with a partner to read through each other's code and **predict** what will happen when they **run** the program.
Review Success Criteria	Display **slide 21**. Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

Lesson 4 – Collision Detection

Aims

- To create a program using a given design.
- To understand the **collision detection** event.

Success Criteria

- Children can plan an **algorithm** that includes **collision detection**.
- Children can create a program that uses **collision detection**.
- Children can read blocks of code and **predict** what will happen when it is **run**.

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

- Princess and the Frog. This is on the main 2Code Page (scroll down to the Chimp activities).
- Princess and Frog algorithm examples –
    - Storyboard Planner (blank).
    - Storyboard Example.
    - Algorithm and Scene Plan.
- Super Coder Poster.
- Blank paper for designing and for Super Coder strips.

Preparation

- Set Princess and the Frog as a 2Do for your class.
- Print and copy Storyboard Planner OR Algorithm and Scene Plan
- Cut blank paper into strips for Super Coder ideas.

Activities

Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.
Recap Events	Use **slide 4** to revise what happened in the Air Traffic Control program at the end of last lesson and introduce today's lesson.
Collision	Display **slide 5**. Show two real-life objects (either real people moving towards each other or objects in your hand) to help explain collision.

	Click though **slide 6** to support your explanation of collision detection.
	Watch the collision detection video on **slide 7.** NB This video is also on the unit main page if required.
Princess and Frog	Display **slide 8**. Work through stages 1 and 2 of Princess and Frog as a whole class. Watch the videos and demonstrate how to get back to the videos (click on the instruction) and unlock the hint. Remind children that unlocking the hint will lose a code star.
The Collision Detection Command	Use **optional slides 9-11** to further explain collision detection if needed.
Activity 1: Princess and Frog – Stage 1&2	Display **slide 12**. Ask children to start the Princess and Frog from their 2Dos area and complete stages 1 and 2 independently.
Turn the Frog into a Prince	Display **slide 13**. Draw children's attention back to the screen, talk through the slide and then return to 2Code and work through stage 3 together – the frog turns into a prince. Ask children: When do we want the frog to turn into the prince? How will we make sure he turns into a prince only if the princess collides with him? (You will need to add this code into the existing **collision detection event**) Demonstrate using the 'image' option in the **action** list (see below). Before clicking on play ask the children to **predict** what will happen when you **run** the program. *Here you could also demonstrate what happens if you put the 'frog image set to prince' **command** outside of the **collision detection** event.
Activity 2: Princess and Frog - Stage 3&4	Display **slide 14**, ask children to independently work through stage 3 and the **debugging** challenge in stage 4.
Your Own Fairy Tale	Use **slide 15**. Questions to deepen understanding: What is the story? What do we want the princess and the frog to do? How will the **objects** move? What will happen if they collide? What will happen next? How will the story end?
Activity 3: Princess and Frog - Stage 5	Display **slide 16**. Ensure that children understand the words in **bold**. Ask the children to complete stage 5 – the challenge stage.

How Did You Get On?	Use **slide 17** to share children's programs, celebrate achievements, and encourage them to evaluate how they got on. Did their program work like the planned **algorithm**?
Super Coder	Display **slide 18** and discuss what children would do if there was no limit to their coding ability (children could write ideas on strips of paper and stick up on your working wall – these could form the basis of future free code lessons).
Activity 4: Extension	**Slide 19** includes a variety of additional and extension ideas. If children complete any of these you could share children's work to a display board (see Appendix 1) and have a 'Coding Show' to share children's programs and celebrate achievements.
Review Success Criteria	Display **slide 20**. Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

Lesson 5 – Using a Timer

Aims

- To understand that **algorithms** follow a **sequence**.
- To design an **algorithm** that follows a timed **sequence**.
- To implement the algorithm in 2Code.

Success Criteria

- Children can create a program that uses a **timer-after command**.
- Children can explain what the **timer-after command** does in their program.
- Children can **predict** what will happen in a program that includes a **timer-after command**.

Resources

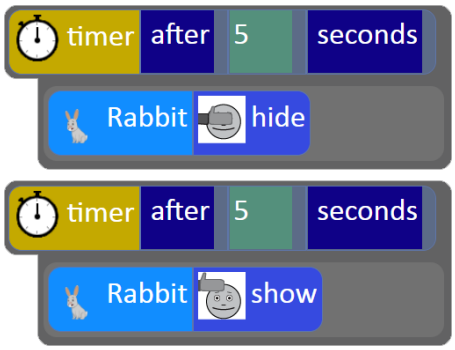
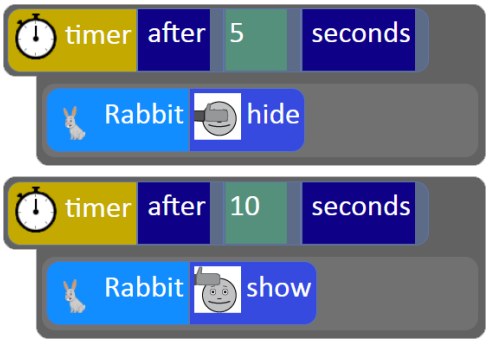
Unless otherwise stated, all resources can be found on the main unit 2.1 page. From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page.

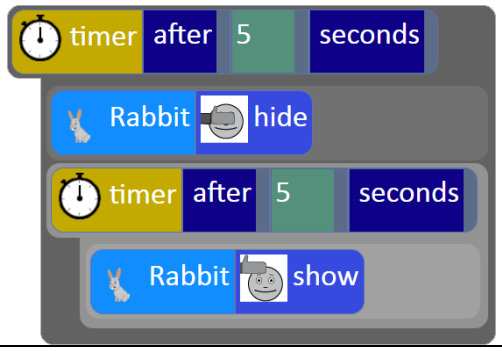
- Magician lesson This is on the main 2Code Page (scroll down to the Chimp activities).
- Storyboard Planner template

Preparation

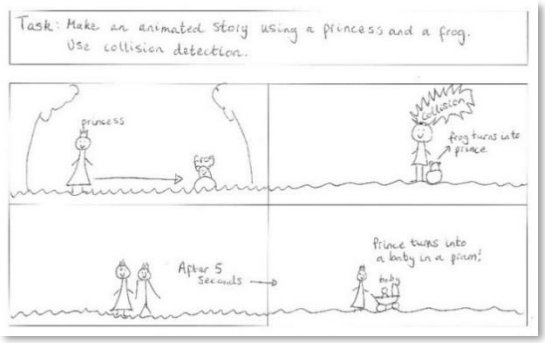
- Set Magician lesson as a 2Do for your class.
- Print and copy the Storyboard Planner for the children. You might want to make this double sided.
- Create a display board for the class to share their programs to. Details of how to do this are given in Appendix 1

Activities

Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.
Vocabulary	Use **slide 4 and 5** to discuss the word '**timer**'.
	Use **slide 6** to introduce the word '**sequence**'.
Sequence of Actions	Display **slide 7** and follow the steps for this offline physical activity to introduce the concept of a **timer.**
	Use **slide 8** to display the algorithm for the sequence. Ask the children to run the sequence and explain that this is called the output. Go through the questions on the slide.
	Display another sequence on **slide 9** and ask the children to 'run' it. Ask the questions on the slide and point out that in both sequences, each child did their action 5 seconds after the last person.
Magician	Work through **slide 10** and watch the video for stage 1 of the Magician guided lesson.
	Work through **slide 11**, then in 2Code work through stage 1 as a class. This is a bit like using an **event** code block – it sets a **timer** and after the specified time the **object** (rabbit) will hide.
	Work through **slide 12**. Complete stage 2 with the class. Add this incorrect code, **test** it and ask the children to help you **debug**:  When you **run** the program both **timers** start at the same time (if you click on stop and play it again you could notice they both highlight orange at the same time) and the code to 'hide' and 'show' the rabbit **executes** at the same time – after 5 seconds. Point out that the **timer** for the rabbit to 'show' needs to start **AFTER** the rabbit has hidden – like the first time when the class timer counted between children.  You need to add the second **timer** *inside* the first **timer** OR work out that the rabbit 'shows' 10 seconds after the start (5 + 5) and alter the second **timer** to reflect that, so either of these solutions would work:

	
Activity 1: Magician	Display **slide 13**. Ask children to start Magician from their 2Dos and work through stages 1-4 independently.
Your Own Magic Code	Display **slide 14** and introduce the stage 5 challenge. Look at the example storyboard plan and look at the different actions that are planned for the objects. Ask children: What do you want the characters to do in your **scene**? What order do you want the **actions** happen in? Explain that the children are going to design an **algorithm** for a sequence of **actions** that are different for the magician and rabbit **objects**. Children could either use the example **algorithm** as the plan for their code **OR** modify the **sequence** of **actions** to create their own **algorithm**.
Activity 2: Magician Challenge Stage	Display **slide 15**, children to return to their 2Dos and complete stage 5. Remind children to save their work continuously as they build up their code.
How Did You Get On?	Display **slide 16**. Ask children to hand in their 2Dos and review their own code – does it do what they planned it to do? What do they need to do to ensure it does?
Activity 4: Extension	**Slide 17** includes an optional extension activity. In the Chimp activities, **'Jumping Monkey'** also uses a timer. (Additional extension activity outlined below).
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.

Additional Extension Activity:
Use Free Code Scenes along with planning templates such as the storyboards to plan a program that uses **timers**.
Encourage Children to think through their designs and annotate them including their confidence in coding what they have designed (red, amber, green), this will give you feedback on areas that Children need help with and help to ensure that Children create realistic designs and successful programs for their skill level.
Example plans are given below.
You could use Example Story program to demonstrate an example (Note: this example uses **key press events** which the children learn in lesson 4.)


Task: Make an animated story using a princess and a frog.
Use collision detection.
princess
frog
collision
frog turns into a prince
After 5 seconds
Prince turns into a baby in a pram!
baby

Lesson 6 – Different Object Types

Aims

- To understand that different **objects** have different properties.
- To create a program using a given **design**.
- To understand the function of **buttons** in a program.

Success Criteria

- Children can create a computer program that includes different **object** types.
- Children can create a computer program that includes a **button object**.
- Children can modify the **properties** of an **object** and a **button** to fit their program design.
- Children can explain what a **button** does in their program.

Resources

Unless otherwise stated, all resources can be found on the main unit 2.1 page. From here, click on the icon to set a resource as a 2do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page.

- Free Code Chimp (this is found on the main 2Code page).
- Snail Race.
- Turtle and Character - Crash Course version
- Road Scene.

Preparation

- Set Free Code Chimp as a 2Do.
- Set Road Scene as a 2Do for less confident children.

Activities

Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.
Objects and Actions	Display **slide 4** and introduce today's topic of objects and actions.
Snail Race	Use **slide 5** and demonstrate by opening the activity from the icon on the slide. Focus on the **actions** available for the snail **object** in stage 1 – have they seen these before? Up until now children have been programming

	objects to move left, right, up, down and stop – but this program works differently. Discuss with them how it is different – snails are a different type of **object** to ones they have used before, and have different options for **actions**.
Different Actions	Look at the scene on **slide 6**. Ask children to predict what will happen when the program is run. Run the program Turtle and Character and see what happens.
Designing a Scene	Display **slide 7** and open Free Code Chimp in front of the class. Follow the instructions on the slide to set the scene.
Choosing Objects	Display **slide 8.** Look at the **object** types to choose from on the left and add a turtle and 3 other **objects** that would move (tell children that in this lesson we are using any object apart from the button – we are going to look at the button later in the lesson).
Changing Objects	Display **slide 9**. Go through how to change the objects and the size of the objects. Recap how to move the objects around. This is the first-time children have used Free Code Chimp in coding lessons so spend a bit of time browsing the clipart galleries – pointing out the categories and search option. Emphasise the final point on the slide.
Activity 1: Create the Scene	Use **slide 10** to challenge children to create a **scene** like this by setting the background and adding **objects** - they could choose different clipart so they all have different **objects** on their scenes. (You could set Road Scene as a 2Do for less confident children so they just have to add objects to the scene rather than create it).
Making Objects Move	Once the majority of children have made their scene draw their attention back to the board and talk through **slide 11.** The **when key** event is picked out, this makes code run when a specified key on the keyboard is pressed (in the example the specified key is the up arrow).
Actions for Objects	Use **slide 12** to re-cap how to program objects to do actions. Point out that the actions for a turtle are different to the other ones. When adding code for the turtle, in this lesson we are going to program it just to move forwards. (Programming a turtle to turn involves some understanding of degrees of a turn – e.g. a quarter turn = 90 degrees, a half turn = 180 degrees). In 2Code open the design you created earlier in the lesson and, with the children, add code to program some of the objects to move. Ask them to predict what will happen, then run the code.
Adding to Your Code	Display **slide 13**. Challenge the children to add their own code to their programs. To give them more support you could return to your program in 2Code, delete the code you had in there and ask children to help you use timers to make the **objects** start at different times. Could they also add some food **objects** in to be eaten along the way? (This will involve using **collision detection**!). **NB:** If children are coding on tablets, the **when key event** is not available. Instead, they could try using **when clicked** or **when swiped events**.

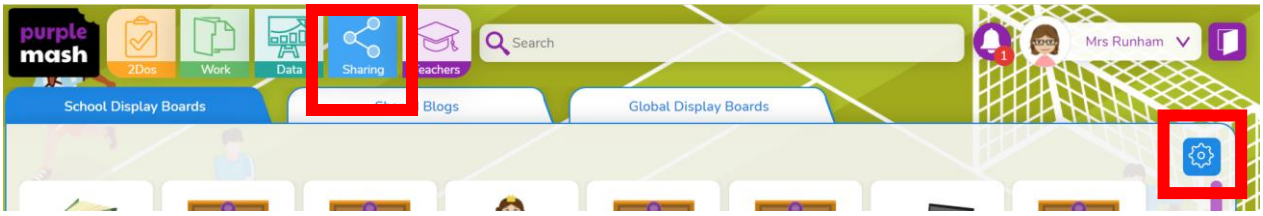
Adding Buttons to Your Code	Display **slide 14** to introduce what a button is and what it does.
Add a Button	Display **slide 15**. In 2Code, add a button to your scene.
Button Properties	Display **slide 16** and look at the button properties. In 2Code, name the button and set the text for the button. You can also change the **text size**, **text colour** and **background** colour of the button.
Coding Your Button	Display **slide 17**. Look at the code blocks and ask children to help you add code to make one of your objects move when the button is clicked on.
Activity 2: Program Your Scene	Display **slide 18**. Set the children back to their own designs to add code to make their **objects** move – challenge them to try using different **events** and add in **collision detection when** a key is pressed. They could also add in **buttons** to make **objects** move. **NB:** If children are coding on tablets, the **when key event** is not available. Instead, they could try using **when clicked** or **when swiped events**.
How Did You Get On?	Display **slide 19** and ask children save their designs. Share great examples with the class, discussing the code that has been used to make them work.
Review Success Criteria	Display **slide 20**. Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

Appendix 1: Display Boards

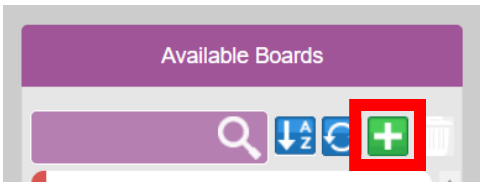
Create the Display Board

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

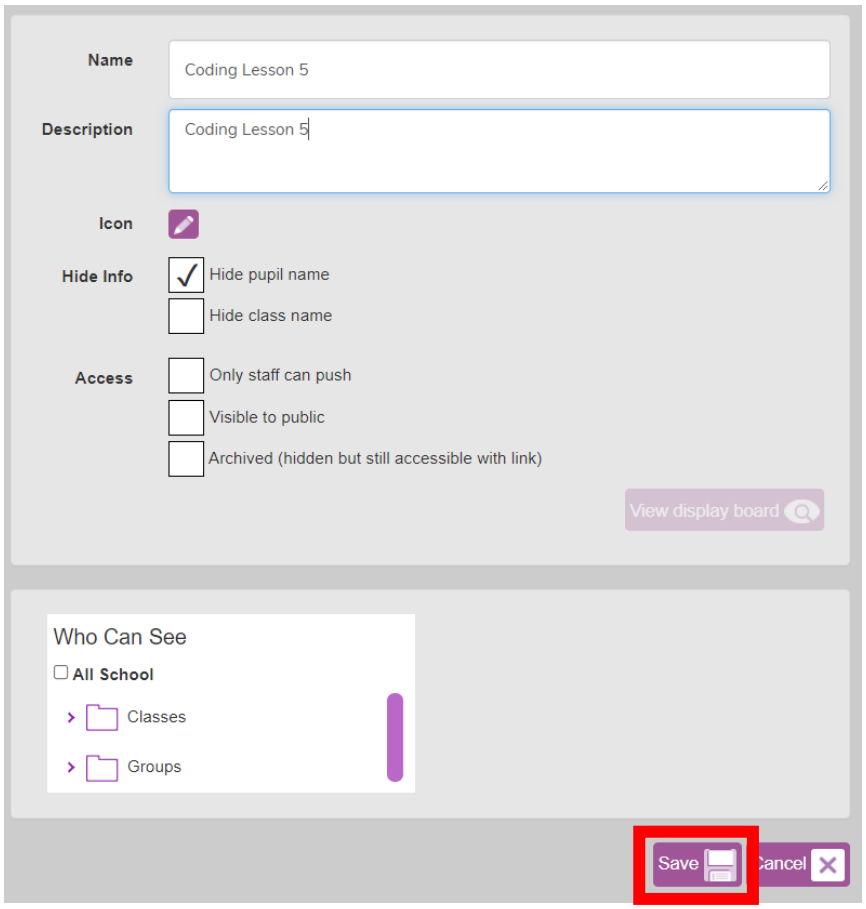
1. Click on the 'Sharing' button to find the Display Board tab, and then click on the settings cog:



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':

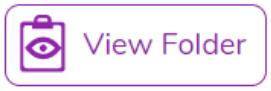


4. Exit Display Board settings:

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:

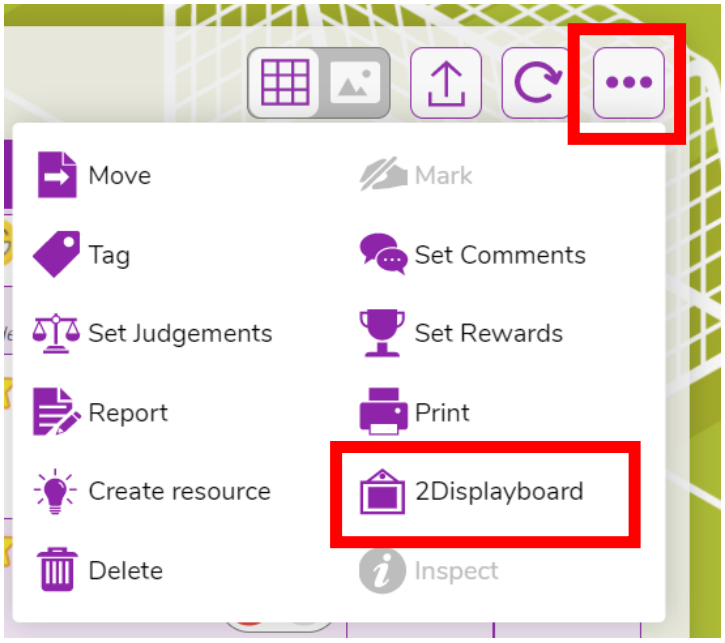
1. Click on 'View Folder' from the 2Do:



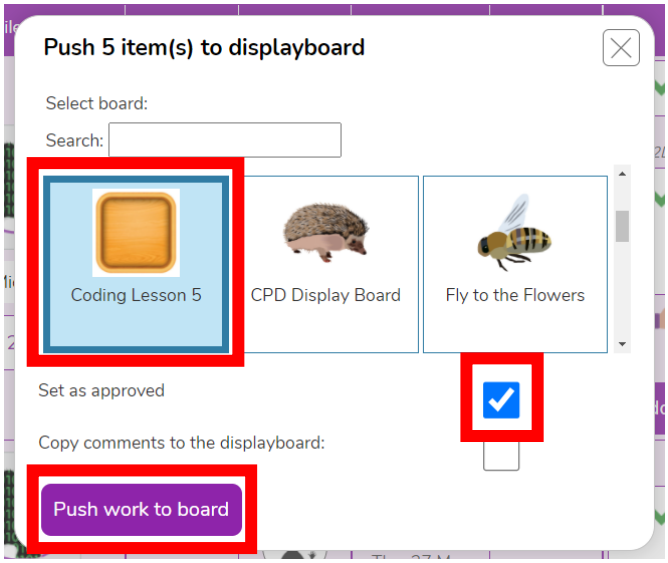
(or navigate to the work you want to share in the Work area).

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':

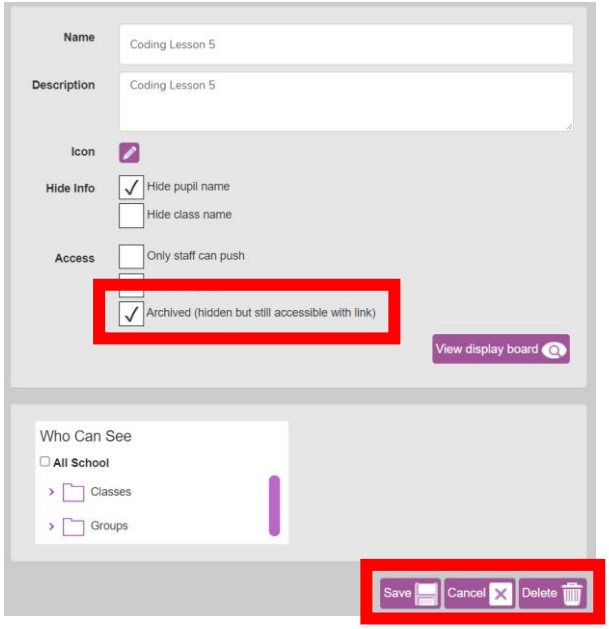


5. Click on 'Sharing' button and then on the display board, you should see the work you've added. It can be deleted by clicking on 'Edit' at the top of the board, then clicking work and then delete. This will remove it from the display board, it won't delete it from Purple Mash.

Deleting or Archiving a Display Board:

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

1. Click on 'Sharing' and then on the settings cog.
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 2 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 have a basic understanding that coding involves writing instructions that a computer can follow. They are developing their understanding that these instructions must be precise and carefully structured through their work in Free Code Chimp making simple one and two step programs for example in the bubble program or making an object move when you click it (lesson 2). Children know that an algorithm is related to giving instructions. They can relate a simple one-step algorithm to the outcome of code in Free code Chimp. For example, in Lesson 3 they have been able to make a program that follows the algorithm e.g. 'when the helicopter is clicked it takes off'. With support, children can create a simple one step program that achieves a specific purpose. With support, children can identify and correct errors (Lesson 6). With support, children can identify the parts of an algorithm that control and initiate specific actions. Based on this, with support, children can predict what will happen in a program (Lesson 6).
Expected	Children have an understanding that coding involves writing instructions that a computer can follow. Children can explain that an algorithm is a set of instructions to complete a task. They have turned algorithms of more than one step into code using free code Chimp. For example, in Lesson 6 they have been able to make a program that follows their algorithm e.g. 'when the animal is clicked it moves forward then turns right'. Children show an awareness of the need to be precise in their designs so that algorithms can be successfully translated into code. (Lesson 3). Children use a planning format on paper before implementing on screen within 2Code as they recognise this is the best approach for designing a solution. They can use the Design Mode within 2Code to carefully see how their planned program will look and are able to switch into Code Mode to apply movements to objects (Lesson 6). They confidently include objects, actions, events and outputs successfully within their 2Code programs. Children can talk through code which contains a timer command, explaining where this command is positioned and what will happen (Lesson 5). Children can predict program outcomes and attempt to debug (Lesson 6). Children can identify the parts of a program that respond to specific events and initiate

Assessment Guidance	
	specific actions. Based on this, children can predict and describe, using a cause-and-effect sentence, what will happen in a program. (Lesson 6). Children can debug their own and other's programs using design documentation to test against (Lesson 6).
Exceeding	Children have a clear understanding that coding involves writing instructions that a computer can follow. Children can explain and give examples that an algorithm is a set of instructions to complete a specific task. They can create complex and logical algorithms of several steps that accomplish the aim of the task that can be easily utilized to create executable code. Children show an awareness of the need to be precise in their designs so that algorithms can be successfully translated into code (Lesson 3). Children can create more complex programs that utilize all the coding constructs that they have learnt about and extend their own learning by trying out different ways to code that achieve a specific purpose. Children can identify and correct errors (Lesson 6). An exceeding pupil will be able to apply their knowledge as a transferable skill across a range of debugging scenarios including making logical attempts to debug their own more complex code. Children can identify the parts of a program that respond to specific events and initiate specific actions. Based on this, children can adopt a systematic approach for predicting the behaviour of programs. Furthermore, using cause-and-effect language, Children can reason in detail about what will happen in a program (Lesson 6).