



Computing

Scheme of Work



Unit 1.7 - Coding



Year Group: 1
Number of
Lessons: 6

From **2**simple



Contents

Introduction	4
Program Design	4
Levels of Scaffolded coding tasks	5
Year 1 – Medium Term Plan.....	6
Lesson 1 - Introduction to coding	7
Aims	7
Success criteria	7
Resources	7
Activities	7
Lesson 2 - Introduction to Block coding on screen	9
Aims	9
Success criteria	9
Resources	9
Activities	9
Lesson 3 Introduction to Backgrounds and Characters	11
Aim	11
Success criteria	11
Resources	11
Activities	11
Lesson 4 - Making a Character Move Left and Right	13
Aim	13
Success criteria	13
Resources	13
Activities	13
Lesson 5 - More actions for characters.....	16
Aims	16
Success criteria	16
Resources	16
Activities	16
Lesson 6 Introduction to Collision Detection.....	19
Aim	19



Purple Mash Computing Scheme of Work – Unit 1.7 – Coding – Contents

Success criteria	19
Resources	19
Activities	19
Appendix 1: Creating a Displayboard.....	21
Appendix 2 - Approving work on a displayboard.....	22
Assessment Guidance	23



Introduction

This unit consists of six lessons that assume no prior coding knowledge. Children will be coding using the 2Code tool.

New 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.

Children will need to be able to drag and drop in order to move code blocks around. If children have not had much practice with this then there are several example activities within the Activities section of 2DIY that help children 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 children have not used Purple Mash before, spend some time showing them how to log in and how to get to 2Code.

The Chimp 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 image and scale in 2Code.

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

Program Design

To master coding skills, children need to have the opportunity to explore program design and put computational thinking into practice. The lesson plans incorporate designing before coding in some lessons.

Storyboarding their ideas for programs. For example, creating a storyboard when planning a program that will retell part of a story.

- Creating annotated diagrams. For example, creating an annotated diagram to plan a journey animation that tells the story of an historical event they have been studying.
- Creating a timeline of events in the program. For example, creating a game program against the computer, what are all the actions needed from the objects?


During the design process, children should be encouraged to clarify:

- the characters (objects and their properties)
- what they will do (actions and events)
- what order things will happen (the algorithm)
- rate their confidence at being able to code the different parts of their design and either refine the design or review possible solutions as a class or group.



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 practice their skills. These are not progressive levels, children can benefit from all the levels of activities at whatever coding skill level they are:

Scaffolding	Task type	Examples of how to provide these opportunities
Most scaffolded  Least scaffolded	Copying code	By giving children examples of code to copy.
	Targeted tasks	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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.
	Project design and code	<p>Projects (imitate, innovate, invent, remix)</p> <p>There are different ways to scaffold learning in projects. This process can be applied to programming projects;</p> <ul style="list-style-type: none"> • 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	<p>Use Free code Gorilla to access the full suite of 2Code objects and commands ✓</p> <p>Use Free code to play and explore freely.</p>

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.

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’.



Year 1 – Medium Term Plan

Lesson	Aims	Success Criteria
1: Introduction to Coding	<ul style="list-style-type: none"> To understand what coding means in computing. To create unambiguous instructions like those required by a computer. To build one- and two-step instructions using the printable code cards. 	<ul style="list-style-type: none"> Children can explain what coding means. Children know that for the computer to make something happen, it needs to follow clear instructions.
2: Block coding	<ul style="list-style-type: none"> To introduce 2Code. To use the 2Code program to create a simple program. 	<ul style="list-style-type: none"> Children can explain what a block of code is. Children can read through combined blocks of code.
3: Backgrounds and Characters	<ul style="list-style-type: none"> To use Design Mode to add and change backgrounds and characters. They will use the Properties table to change the look of the objects. To use the Properties table to change the look of the objects. 	<ul style="list-style-type: none"> Children can make a background using Design Mode. Children can add characters using Design Mode. Children can use the drop-down menu to change backgrounds and characters.
4: Moving characters	<ul style="list-style-type: none"> To design a scene for a program. To use code blocks to make the characters move automatically when the green Play button is clicked. To add an additional character who moves when clicked. 	<ul style="list-style-type: none"> Children can design a simple program and then create the program using 2Code. Children can write a program that controls how a character will move. Children can make a character move when clicked.
5: More actions	<ul style="list-style-type: none"> To explore the When Key and When Swiped commands (on tablets if available). To use the Stop button to make characters stop when the background is clicked. 	<ul style="list-style-type: none"> Children can program a character to move given a variety of input events.
6: Collision Detection	<ul style="list-style-type: none"> To explore a method to code interactivity between objects. To use Collision Detection to make objects perform actions. To use the sound property. 	<ul style="list-style-type: none"> Children can use collision detection to make objects interact. Children can program a sound to play when objects collide.



Lesson 1 - Introduction to coding

Aims

- To understand what coding means in computing.
- To create unambiguous instructions like those required by a computer.
- To build one- and two-step instructions using the printable code cards.

Success criteria

- Children can explain what coding means.
- Children know 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 [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 don't navigate away from this page.

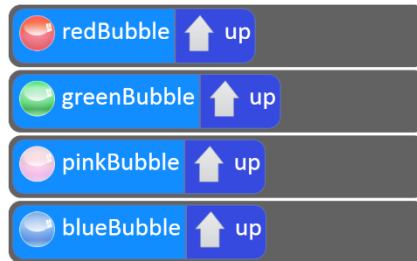
- [Code block cards](#). Children will need to use a few copies of each picture to create code away from the computer.
- Chimp Bubbles Guided activity. This is on the [main 2Code Page](#) (scroll down to the Chimp activities).
- (Optional) Exercise books to be used as 2Code workbooks for recording coding exercises, if desired.

Activities

1. Explain to the children that we are looking at **coding**. Ask them if they know what coding is. Discuss briefly that it is the way that computer programmers **input** instructions into computers to create programs. Can they give any examples of computer programs that they have used?
2. Start off by doing some activities where the children must follow or give clear instructions.
 - Choose two children; one is a robot and the other is a coder. The coder needs to direct the robot to walk from one place in the classroom to another. How can they give the instructions so that the robot does not crash into objects in the way? Repeat a few times in different locations.
 - Tell the children that you are now going to be the robot and they are the coders. Stand by the whiteboard and ask the children to give you clear instructions, one step at a time, for drawing a basic picture of a house.
 - Once you have a set of clear instructions, discuss the way that coding languages use symbols rather than whole sentences. Can they come up with their own symbols for the instructions for drawing a house? For example, holding their fingers in a triangle shape for the roof. These symbols could be written on the whiteboard to create a program. This can become more complex, depending upon the understanding and interest level of the children. Some children will enjoy working out how their 'code' could be adapted to draw a bungalow or a block of flats. Some children will want to be precise about the placement of the shapes, e.g. a symbol to show putting the roof *on top of* the house.
3. Show the children the printed **code block** cards. Explain that these are examples of code used on a computer. Can they suggest ways to combine the cards to make instructions? Show them an example of combining the cards; you could show the children the first stage of the Bubbles activity in 2Code Chimp in which a bubble goes up.



- Pair up the children and give the pairs some printed out code block cards. They should join two blocks to give one clear instruction (i.e. red bubble – up). 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.





Lesson 2 - Introduction to Block coding on screen

Aims

- To introduce 2Code.
- To use the 2Code program to create a simple program.

Success criteria

- Children can explain what a block of code is.
- Children can read through combined blocks of code.

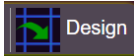
Resources

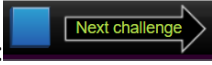
Unless otherwise stated, all resources can be found on the [unit main page](#). From here, they can be set as 2dos by clicking on the icon. To preview resources linked to here, right-click and 'open in new tab' so you don't navigate away from this page.



- Code block cards from Lesson 1.
- [Fun with Fish Activity](#). This is on the [main 2Code page](#) in the Chimp section.
- Bubbles Activity. This is on the [main 2Code page](#) in the Chimp section. Set this as a 2Do for the class.
- Optional: Exercise books to be used as 2Code workbooks for recording coding exercises and designs.

Activities

1. Remind the children about the blocks of code that they were using last lesson. Explain that today they are going to be coding on the computers using blocks of code.
2. Show children the Fun with Fish lesson.

3. Briefly show the Design View button  (in the top right-hand corner) to show the look of the program and the **object** that the code will be controlling (the fish).
4. Complete step 1 as a class; emphasise the need to give the computer clear instructions for moving the fish. The available **actions** for the fish object pop-up as soon as the fish is dragged into the code window. Show the children what to do if they click on the wrong direction → click on the direction again and select the correct one.

5. Show the children where the Play button is to **run** the code and how they can see to move to the next stage of the activity or stop the code running to make changes: .

6. Show the children how to replay the instruction video by clicking on the  button and how to get an extra hint by opening the video and clicking on the  button.

7. Complete step 2 and move onto step 3. Explain that this is a step where you must fix the code that the monkey has got wrong. Complete this as a class. Move on to the next step; you will firstly be told how many monkey stars you have got. The maximum is 5; you lose stars for using hints.
8. Step 4 is the Challenge step. All the guided activities have this challenge step, and this is where children deepen their understanding of the code that they have been working on. Take a few

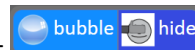
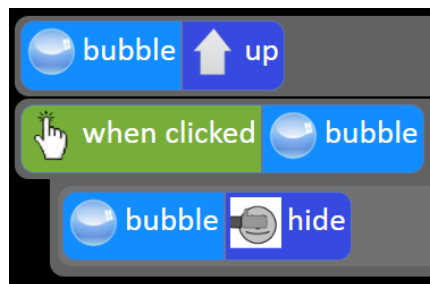


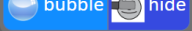
suggestions from the class about how to improve the fish tank including how to use the

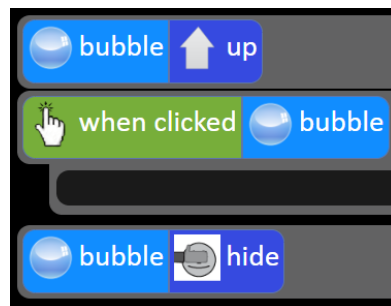


command and how to write the code to do this.

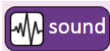
9. Show the children how to save their code in their own My Work folder giving the work a sensible name. Talk about why a sensible name is important.
10. If the children have not used Purple Mash before, spend some time showing them how to log in and how to get to 2Code.
11. Direct the children to the Bubbles activity, either via their 2dos or via the main 2Code page. Ask them to complete Step 1 and try step 2.
12. Bring the children back together to discuss the solution to step 2 and check that all children could complete it. Read the solution together: 'When the bubble is clicked, the bubble should hide.'



13. Explain that each action is on a different line. Point out that  is slightly indented. This is because it is the **output** (what we want to happen). Show what would happen if you put the codes directly under each other. **Run** this code and talk about what happens.



14. Explain that the computer is confused. We are asking it not to do anything when we click the bubble and also for the bubble to hide when the code is run.
15. Children should try to complete the rest of the bubble activities on their computers including the challenge. Draw their attention to the naming of the different coloured bubbles – if they were all called 'bubble' then it would be very confusing. It is important to give each **object** a sensible, useful name.
16. Look at some of the challenges. Has anyone managed to go into Design view and add more bubbles?

Has anyone **tinkered** with the  command?



Lesson 3 Introduction to Backgrounds and Characters

Aim

- To use Design Mode to add and change backgrounds and characters.
- To use the Properties table to change the look of the objects.

Success criteria



- Children can make a background using Design Mode.
- Children can add characters using Design Mode.
- Children can use the drop-down menu to change backgrounds and characters.

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 don't navigate away from this page.

- [Free Code scene cards](#). Print copies of these, children will be recreating some of them.
- [Challenge cards](#) Print copies of these, children will be recreating some of them.
- [Free code Chimp](#); this is accessed through the [2Code main page](#) in the Chimp section. Alternatively, you can set it as a 2do for the class.
- Save an image file suitable for a code background; a nature scene or city scene or even a photo of your school somewhere that you can access from your device – this is to demonstrate in step 7.

Activities

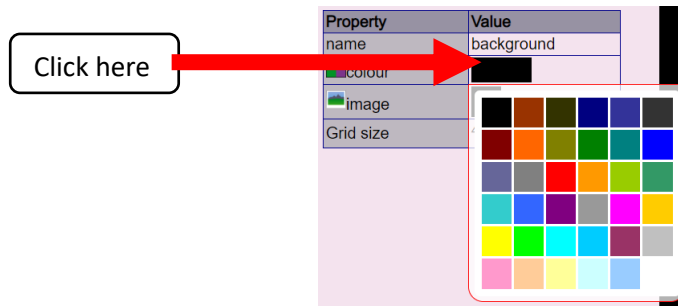
1. Show the children Free Code Chimp. This is different to the Chimp lessons because you have the freedom to create your own **programs** and you can **design** the look and add the **objects** to the program.
2. Explain that the children are going to be programmers and that their task is to create their own simple game. Discuss any computer games that the children have played. On the interactive whiteboard you could show images of some popular age-appropriate computer games.
3. Before starting the code, you need to think about how your game will look. This is called the **design stage** of the coding process. Refer again to the pictures of popular games. Just like pages in a book, a game needs to have a background image. What else do games need? Discuss the use of **characters/objects** in games and the requirement for a game to have an aim. For example; to collect items, to save other characters, to score points or to get onto another level or into another world.
4. Remind the children about Design View and the Code View in 2Code that they saw last in the previous lesson. To switch to Design view, click  Design. When in Design view, click  Exit design to switch to code view. Currently, the design is blank. This is because we haven't chosen a **background** or any **objects**.



5. Demonstrate the following in Design View: To add a background, press the Background button. An information box will appear that has the **properties** of the background:

Property	Value
name	background
colour	
image	
Grid size	4

6. To change the colour of the background, you can click on the colour property and select a different colour:



7. To choose an image for the background, click on the next to the 'image' title. Show children how to choose different backgrounds including the Camera (if you have webcams), Choose file and Paint buttons.
8. Next, show children how to add a character to their background by dragging the character box from the left-hand side into the design.
9. Show children how to change the character image **property** by double clicking on the character and using the drop-down menu of the clipart picker.
10. Show children how to change the character scale **property** by clicking the scale property in the property box on the left and using the green arrows to increase or decrease it. What effect does this have on the character?
11. Explain that you have a selection of designs for games that need finishing and show the children some of the scene and challenge cards. Children need to recreate the scenes and then follow the design requests detailed below them. Decide how you wish to distribute the cards.
12. Remind children how to save their work.
13. Using the Free Code scene cards and challenge cards, children should recreate the scenes using Free Code Chimp, choosing the correct backgrounds, characters and properties.
14. Once children have created a few of these they might wish to create their own scene for a game to be continued next lesson.



Lesson 4 - Making a Character Move Left and Right

Aim

- To design a scene for a program.
- To use code blocks to make the characters move automatically when the green Play button is clicked.
- To add an additional character who moves when clicked.

Success criteria


- Children can design a simple program and then create the program using 2Code.
- Children can write a program that controls how a character will move.
- Children can make a character move when clicked.

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 don't navigate away from this page.

- [Year 1 vocabulary quiz](#).
- (Optional) Flashcards from [Pupil Flash Cards](#) and [Teacher flash cards](#). The Teacher flash cards have been created in such a way that you can print them on A4 paper, cut them to size, fold them in half and glue them together.
- [Printed storyboard template](#) – alternatively, children could draw their designs in their workbooks or on plain paper

Activities

1. Use the quiz as a class. It is set up so that you attempt all three questions and then click the  button to check the answers. Click 'OK' to see which are correct and incorrect:

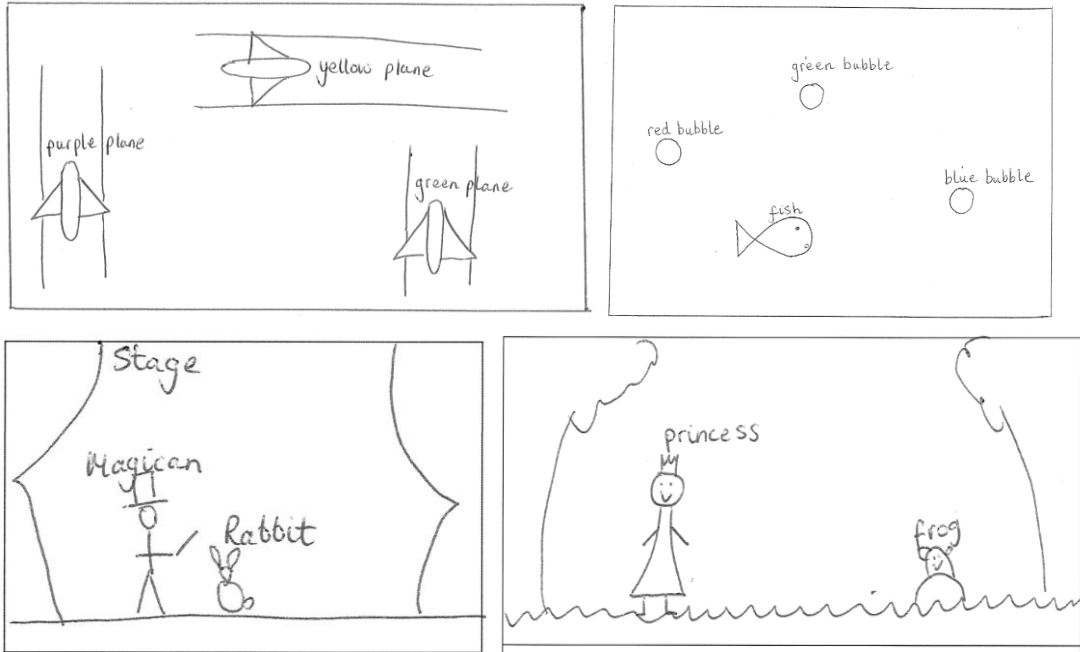
Children will have encountered most of the vocabulary during the previous lessons. You can use the vocabulary cards to find the answers and display in the classroom.

2. Explain that children will be creating their own scenes today. They could use the ideas that they started last lesson or choose new scenes. Look through some picture books that show different

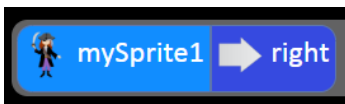


backgrounds and characters. Discuss how the characters are relevant to that scene. For example, you have a snowman in a winter scene and not on a beach.

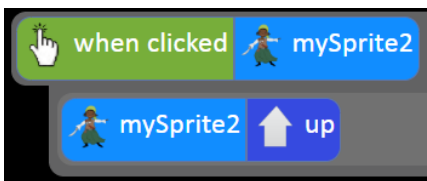
- Hand out the storyboard templates (or alternative). Children should work on a basic idea for the type of background and no more than three characters. If they are using the 4-box storyboard, they should fill in just the first box which will be how the program looks initially. Drawing should be basic e.g. labelled stick figures. Here are some examples:



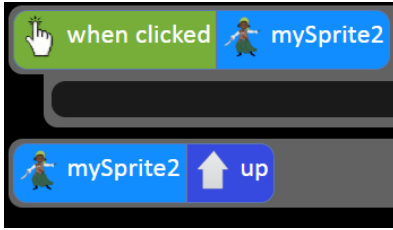
- They could review their designs in pairs and discuss the types of scenes they would like to create. They could they be linked to their favourite books, plays, films etc.
- Next, model how to add code to the character in code mode by dragging in the appropriate blocks to make the character move left or right. Here is an example:



- Remind the class how to run the code. Check that the code works and then show the children how to make another character move when clicked:

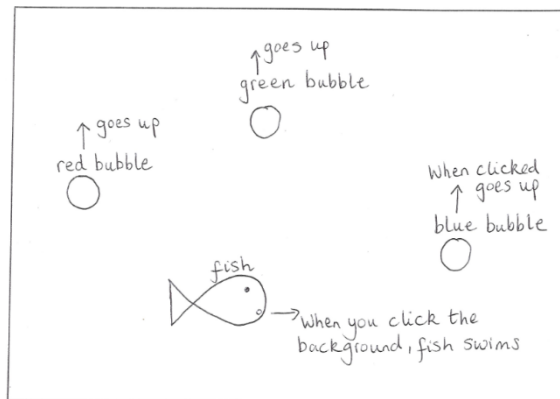


- Children might have done this in the Bubble activity. Remind them that each action is on a different line and the outcome from clicking on the object is slightly indented. Can they remember what would happen if they wrote the code like this:



The computer is confused. We are asking it not to do anything when we click mySprite2 and also for mySprite2 to go up when the code is run.

- Children should add an arrow to one of the characters to show the direction that they are going to code them to move when they click play. They should add an arrow and the words 'when clicked' to a character who will move when clicked. They could add some movement for all the characters. Here is an example:



- Once they have completed their design and are confident that they can code it, they should create the program in 2Code and save it.
- Once children have completed and tested their code, they could refine their design to add more complexity and then attempt to code it by tinkering with 2Code.



Lesson 5 - More actions for characters

Aims

- To explore the When Key and When Swiped commands (on tablets if available).
- To use the Stop button to make characters stop when the background is clicked.

Success criteria



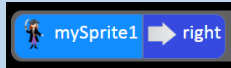
- Children can program a character to move given a variety of input events.

Resources

- Create a displayboard for children to share their work to. See [Appendix 1](#) for details of how to do this.

Activities

- Ask children if they remember what the following terms mean and if they can give an example of one; an object, an action, a command, an event. Here are some possible answers:

	Meaning	Example
an object	An element in a computer program that can be changed using actions or properties.	A character. 
an action	Types of commands, which are run on a particular object. They could be used to move an object.	Moving e.g. making the character move right. 
a command	A single instruction in a computer program.	A line of code e.g. 
an event	This happens when an object responds to either the user creating an input (clicking, swiping etc) or another object. NB Children have only experienced when Clicked thus far.	'When Clicked' is an event that happens when the object is clicked.

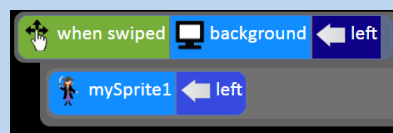
- Explain that in this lesson we don't want the characters to move as soon as the code is run. We want them all to move after an event has occurred. Ask for children's suggestions as to how we could achieve that. Try to get them to think of ways that they move characters in any games they play on devices, i.e. when you press a button a character moves, using a joystick etc.
- Open Free Code Chimp and ask a child to add a background and a character on the whiteboard. Can the child remember how to make the character move when clicked on (the when clicked event)?
- Look at the other Event blocks on the left-hand side. Ask the children whether any of them have had a chance to tinker with these. They could come up to the board and show the class how to add a



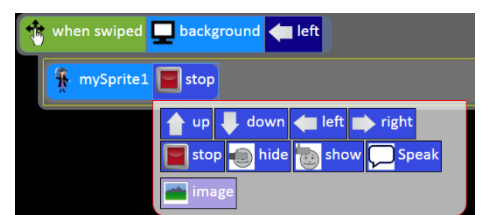
character who responds to a different event. **Note:** the available events will vary depending upon the device type. A PC will have ‘when key’, ‘when clicked’, ‘when swiped’ events. A tablet will have ‘when clicked’, ‘when swiped’ and ‘when tilted’ events. If you are demonstrating on a computer and children are using tablets, then be aware of this.

- Discuss the difference between the event types for computers and tablets. Can children work out that making the keyboard appear on the tablet would make the screen tiny and unusable? Adding a ‘when tilted’ command for a PC would not work as the PC has no movement sensor to detect it being tilted (and it would be very awkward to do!)
- Demonstrate how to use the different event commands depending upon their device type.

	Questions to ask	Things to try
When clicked	What sorts of things would you want to happen when the player of a game clicks?	Making one character do something when a different character is clicked Making a character stop when the background is clicked
When key	Have children played games where they must press certain keys to move a character?	This is how you can code for two players playing different characters e.g. a car race game. Pressing the arrow keys to make one-character move and different keys to make another character move. Writing code for when the space key is pressed.
When swiped	Have you played games that use swiping? Is it more common on a PC or on a tablet?	Using swipe command for when a character is swiped is tricky because once they are moving it is hard to swipe in the right place to get them to change direction. Try writing code so the character moves when the background is swiped.
When tilted	Have you played games that use tilting? What did the tilting do?	Can they make a game where they tilt the tablet to get a ball in a hole or to a target? (They will only be able to code the movement at the moment).



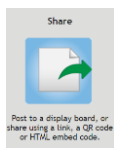
- Each time, ask children to ‘read’ the code e.g. ‘When you press the U key, my character moves up the screen.’
- Explain that when we press the When Clicked or When Swiped buttons the character moves and it keeps on moving. How can we make the character stop?
- Have a look at how the Stop command works; this is an available action once a character has been dragged into the code window.





10. If the children write code that makes a character move when clicked, they cannot also have code that makes the character stop when clicked as they are asking for different outputs to happen for the same input. They will need to call the stop action when a different event occurs for example, when the background is swiped down.
11. Ask the children to decide upon some events to try in a new program (or they could add to the program that they started last week). They could make a note of what they are going to explore to keep them on track and make a design diagram. Show them how to open their work or open a new free code Chimp file.
12. Give children time to make simple program.
13. Remind children how to save and then show them how to share their work to a displayboard:

14. Click on the share button  (they must have saved their work first).



15. Click .
16. Find the displayboard that you created for them and click on it and then 'ok'.
17. You (the teacher) will then need to bulk approve all the work so the children can see it. See [Appendix 2](#) for a guide to how to do this.
18. Show some of the children's work using the Displayboard. The child whose work it is, should read their own code to explain what the program should do and you can try it out.



Lesson 6 Introduction to Collision Detection

Aim

- To explore a method to code interactivity between objects.
- To use Collision Detection to make objects perform actions.
- To use the sound property.

Success criteria

- Children can use collision detection to make objects interact.
- Children can program a sound to play when objects collide.

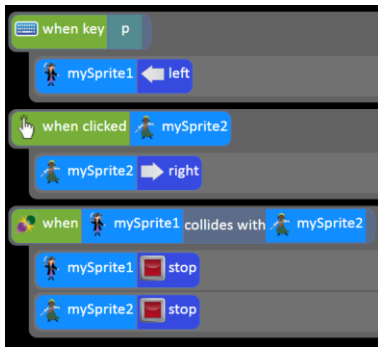
Resources

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 don't navigate away from this page.

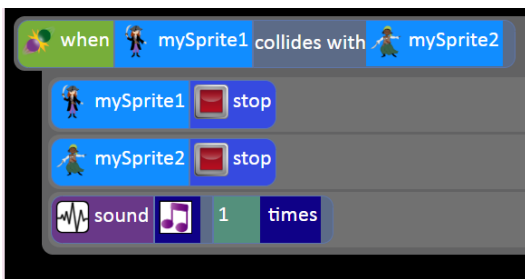
- Vocabulary flash cards from the previous lesson.
- [Collision Detection video clip](#) for use with the class.

Activities

1. Open Freecode Chimp on the whiteboard and review the events blocks that children have used.
2. Show two real-life objects (either real people moving towards each other or objects in your hand). Are the objects moving in the same direction? What will happen when the objects meet? Explain the meaning of the word 'collide'. Can children see this word in 2Code?
3. This lesson they will be learning how to make objects interact with one another by writing code for what happens when they collide.
4. Ask a child to come up to the whiteboard, add a background, two characters and change the characters' images to one from the available clipart in Design Mode.
5. Save the file and remind children how to do this. Why is it important to save before you have finished writing your code? Discuss (a) not losing work and (b) having a clean version to go back to if you mess up your code and want to go back a step to a working version.
6. Watch the collision detection portion of the video; **Collision Detection is discussed at 2.00 mins remaining of this screencast.**
7. Show how to insert the collision detection event in 2Code. Drag it into the code window and point out the two question marks to fill. Ask children to speak some sentences filling the two question marks with objects; these could be the ones on the screen or any 'real' object e.g. 'When the car collides with the wall'.
8. Add code so that the objects stop moving when they collide:



9. If you only added one object to your code, would you be able to use the collision detection command?
10. Direct children’s attention to the sound output command. What do they think this does?
11. Drag in the sound command so that it makes a sound when the objects collide. Click on the question mark to choose an appropriate sound.
12. Show children that they can select a sound from the sound picker.



13. Children should now work on the programs they created in the previous lessons to add code for Collision Detection. This to make the objects stop moving and make a sound.
14. Encourage children who have finished to investigate the other options within the sound picker; the Record and Piano buttons. How do these functions work?
15. Children experiment with adding suitable sounds upon collision. Children can also experiment with making an object hide upon collision (e.g. a spaceman moving towards a rocket: upon collision, the spaceman hides, making it look as if he went into the rocket).
16. Chimp lessons ‘Guard the Castle’ and ‘Princess and the Frog’ also use Collision Detection.




Appendix I: Creating a Displayboard

For detailed information about Purple Mash Displayboards, see the [User Guide](#) in the [Teachers>User Guides section](#) of Purple Mash. These instructions tell you only the steps needed for the displayboard this unit of work



Click on the Admin tab and select the [Manage Display Boards](#) icon to access the Display Boards control panel.

You will see a list of the existing boards. If this is your first board the list will be empty.

Under Available Boards click the  icon and choose a name for your board.

You can optionally add a description for the board and choose an icon to represent your board.

For this board, do not tick any of the Hide Info or Access tick boxes.

In the bottom section, locate your class by clicking on the arrow next to the Classes folder and ticking the box by the class name.

Click the Save button at the bottom of the screen to save your board.

You're done! Your board is ready to receive new projects from pupils.

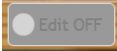
NB You will only be able to add classes if you are the allocated teacher for that class.

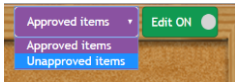


Appendix 2 - Approving work on a displayboard


For detailed information about Purple Mash Displayboards, see the [User Guide](#) in the [Teachers>User Guides section](#) of Purple Mash. The steps below assume that you are going to bulk approve all entries on the displayboard as part of lesson 4.

Open the displayboard from the home screen by clicking on the Sharing tab and then on the displayboard.


Turn on editing by clicking the  slider. Then select 'unapproved items' from the drop-down list.



Click on the first item on the page, click and hold down, the Shift key, then click on the last item on the page.

Then click the  icon to approve selected projects.

If there are more pages (grey arrow on the right), go to these and approve the work in the same way.

To return to the Display Board main screen, click the  icon.



Assessment Guidance

The unit overview for year 1 contains details of national curricula mapped to the Purple Mash Units. The following information is an exemplar of what a child at an expected level would be able to demonstrate when completing this unit with additional exemplars to demonstrate how this would vary for a child with emerging or exceeding achievements.

Assessment Guidance	
Emerging	<p>Children have a basic understanding that coding involves writing instructions that a computer can follow.</p> <p>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 lesson 2 bubble program or making an object move when clicked on in (Unit 1.7 Lesson 4).</p> <p>With support, children can create a simple one step program that achieves a specific purpose. In (Unit 1.7 Lesson 2), they can make a bubble object move.</p> <p>In (Unit 1.7 Lesson 3), they can create a scene with support. In (Unit 1.7 Lesson 4), they can make character objects move. In (Unit 1.7 Lesson 5), they can make a character move when clicked but might not be able to plan how to make a character move when a different character (or the background) is clicked.</p> <p>Children are beginning to understand that they can correct unexpected outcomes by changing the code and they make attempts to identify the source of bugs.</p> <p>With support, children can explain the possible actions of objects including movement, clicking on them and collision. When looking at a simple program they can ‘read’ the code one line at a time but might not be able to envision the bigger picture of the overall effect of the program. Children will be able to suggest that an object might move when clicked but might not be able to suggest that an object might move when the background is clicked.</p> <p>With support, children can manipulate how their program looks using the 2Code design mode, by adding and changing backgrounds, characters (Unit 1.7 Lesson 5. Point 3), sounds (Unit 1.7 Lesson 6. Point 12) and objects. They can create a program that controls a character.</p>
Expected	<p>Children can both give and receive verbal instruction to achieve a simple outcome such as getting from one point of the classroom to the other whilst avoiding obstacles (Unit 1.7. Lesson 1 Point 2). Furthermore, they can use printed block-based code to also articulate a simple set of instructions (Unit 1.7. Lesson 1 Point 4). Children can apply off-screen block code to on-screen block code within 2Code.</p> <p>Children can consider a variety of factors when coding, including the way that the program is designed.</p> <p>They can then design programs that control the look and the actions of objects.</p> <p>Their designs show that they have thought about the need for precise, purposeful, ordered instructions. For example, (Unit 1.7 Lesson 4. Point 8), they consider the kinds of actions they know to be possible when designing their program with a partner.</p> <p>In (Unit 1.7 Lesson 5), they think about the program they are making with reference to the objects, the actions and the output e.g. they know that an object will get clicked on and then an object will do something in response.</p>



Assessment Guidance

	<p>They can then construct their code purposefully to make objects interact. Using the 2Code design mode, children can manipulate how their program looks by adding and changing backgrounds, characters (Unit 1.7 Lesson 5. Point 3), sounds (Unit 1.7 Lesson 6. Point 12) and objects. they can break a problem down into small chunks and then combine it to see an outcome e.g. combine two parts of code “When we click the red bubble, red bubble hides.”</p> <p>They know that any unexpected outcome is due to the code that they have created and make logical attempts to try to fix this code rather than attributing it to a fault with the computer understanding the instructions.</p> <p>When looking at a program they can ‘read’ the code one line at a time and make good attempts to envision the bigger picture of the overall effect of the program. They understand that new ‘actions’ within instructions (block coding algorithm) are on a new line (Unit 1.7. Lesson 2. Children will be able to suggest that an object might move when clicked or suggest that an object might move when the background is clicked.</p> <p>Furthermore, children consider the end user of their program and make purposeful changes to the user interface to enhance functionality (Unit 1.7 Lesson 5. Point 5).</p> <p>Most children will be able to save their 2code files, using a memorable file name, to their own personal space on Purple Mash and understand that this can be retrieved later Unit 1.7 Lesson 2 Point 9.</p>
Exceeding	<p>Children can consider a variety of factors when coding including the way that the program is designed.</p> <p>They can then design programs that control the look and the actions of objects and their interactions with one another.</p> <p>Their designs show that they have thought about the need for precise, purposeful, ordered instructions. For example, in (Unit 1.7 Lesson 6), they can combine the knowledge of moving objects, clicking on objects and collision detection to create their own multi-line program.</p> <p>Children design their code purposefully and consider a variety of factors when coding including the way that the program is designed.</p> <p>They can then code more complex programs that control the look and the actions of objects and their interactions with one another including click events and collision detection.</p> <p>They intuitively debug their code knowing that any unexpected outcome is down to the code and not the computer’s understanding.</p> <p>Children can explain the possible actions of objects including moving, responding to being clicked on and collision with other objects. When looking at a program they can ‘read’ the code one line at a time and take account of the nesting of lines of code inside each other e.g. the action of a click event inside the click event. They can also use their prior coding experience to recognise whole blocks of familiar code instead of reading line by line which helps them to envision the bigger picture of the overall effect of the program.</p> <p>Using the 2Code design mode and the properties table, children can manipulate how their program looks by adding and changing backgrounds, properties of characters (Unit 1.7 Lesson 3. Point 8) (Unit 1.7 Lesson 5. Point 3), sounds (Unit 1.7 Lesson 6. Point 12) and objects. They can create a program that controls a character. Furthermore, children consider the end user of their program and make purposeful changes to the user interface to enhance functionality (Unit 1.7 Lesson 5. Point 5). Children can share their programs using shared display boards.</p>