purple mash

2Code User Guide

2simple

Contents

1 Introduction

What is 2Code?

2Code is a tool to introduce computer programming (coding) to children.

2Code has three key components: free code, guided lessons and debug challenges. See the following sections for brief details of each.

Free Code

Guided Lessons

Debug Challenges

Planning your coding lessons

The place to go when planning your coding lessons is the Teacher section of Purple Mash. You need to be logged in as a teacher to access this area.

There is a **Computing Scheme of Work** which has a coding unit in every year group which uses the free code component of 2Code as well as some of the guided activities. Plans for this can be found by clicking the Computing Scheme of Work button in the Teacher section or with this link. For details of what is covered by these units see the section of this guide called Free Code.

There are also **guided lessons** that use 2Code. The 2Code Guides and Resources section of Purple Mash contains links to the lesson objectives and solutions for these. For further details see the Guided lessons section of this guide.

The 2Code Guides and Resources section of Purple Mash also contains flashcards, certificates, and a link to the 2Code glossary.

Pupil 2Code area

2Code is accessed by pupils in the Tools section.

The 2Code Tool area contains:

- Tutorial videos
- The Guided activities spilt into stages
- Free code links for each stage
- Vocabulary quizzes

- Games

Curriculum Maps

Curriculum maps organised by country and can be found on the main teacher page in Purple Mash.

2 Getting started

2Code itself is spit into three complexity levels. This applies to both the guided lessons and free code. This section contains information about what is included in these levels.

How 2Code is structured is an introduction to the toolbars and layout for all levels.

- Chimp includes the functionality of the Chimp level.
- Gibbon includes the added functionality of the Gibbon level.
- Gorilla includes the added functionality of the Gorilla level.

This section also contains information about:

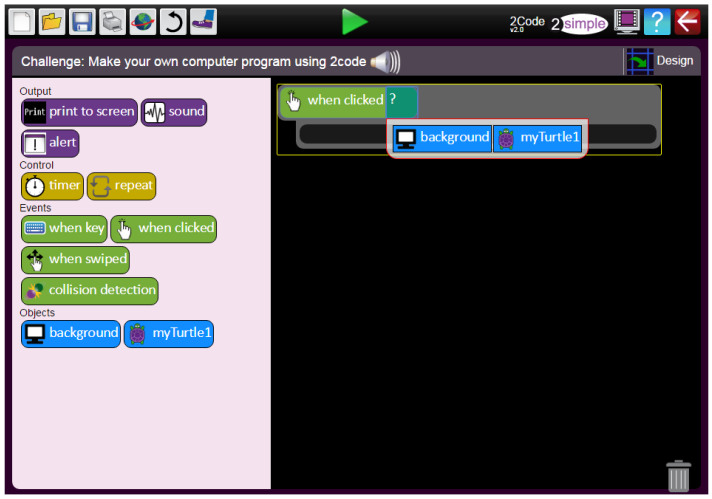
- The glossary
- The colour key
- Sharing 2Code programs
- Real code mode

2.1 How 2Code is structured

2Code uses block coding to build up programs. This means blocks of code are dragged by the user onto the coding window and they fit together to build up the program.

When a block of code is placed in the coding window, 2Code then offers the user a choice of appropriate functions to complete the line of code.

The following picture shows the Chimp level.



On the left-hand side are all of the commands, the available commands depend upon the level.

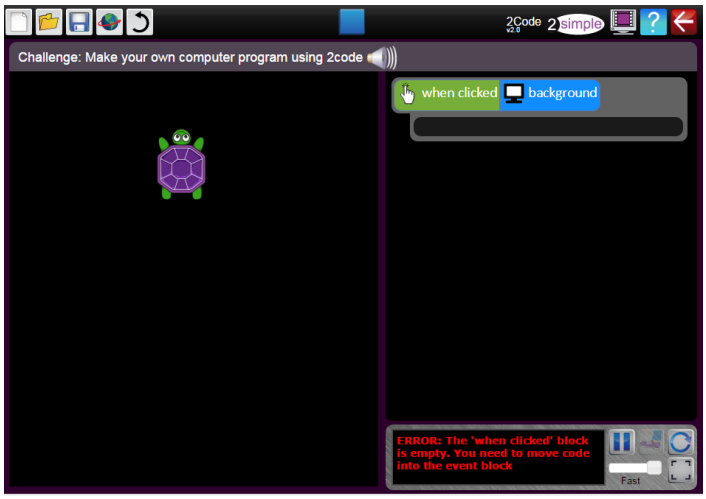
The main section in the middle is the main code window.

Code is written code by dragging blocks. An orange highlight will indicate where the command will go in the code window, this helps you to check that the command is in the correct place.

Delete code by clicking the block to delete and then clicking on the bin in the bottom right-hand corner or by dragging the code to the bin.

Once the code is placed, 2Code will indicate the next area to be coded. In the example above, the user has dragged the 'when clicked' block and now needs to select the thing to be clicked; the background or the turtle.

To run the code, press the Play button at the top centre of the screen.



In Play mode, a debug window appears at the bottom right of the screen, this highlights errors and enables you to use the debug tools. See the debug section for further details of these.

2.1.1 Toolbar

The toolbar at the top contains the familiar functions:

 New file/restart lesson

 Open file

 Save file

 Print; print code as a PDF.


Undo


Step through the code - see the step mode section.

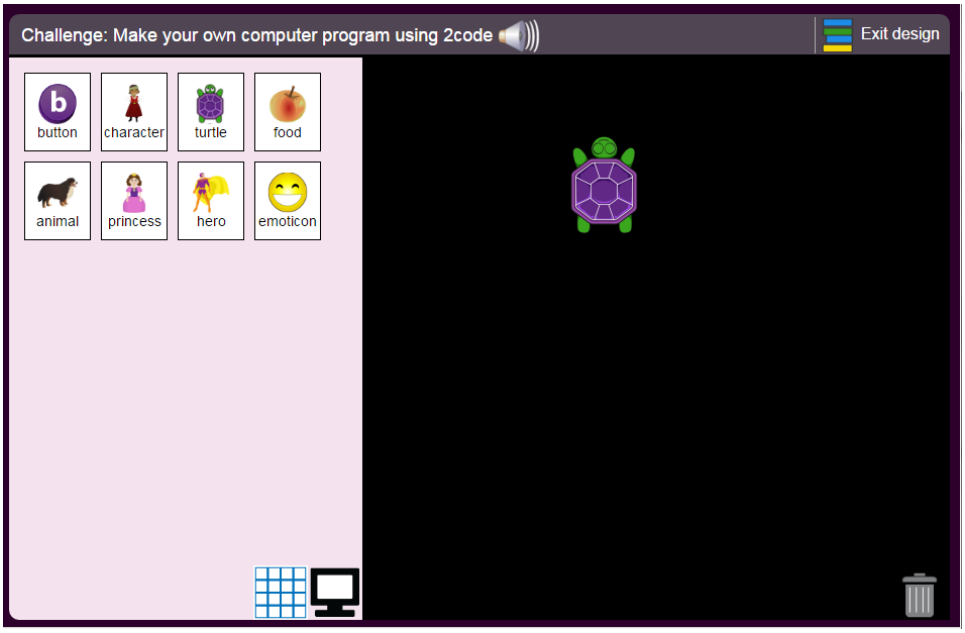

Share: when code is shared it will open in full screen mode so viewers can play the code but cannot change it.


Video Tutorials.

2.1.2 Design view

The look of the program that you are coding is designed in design view.

To open design view click on the 



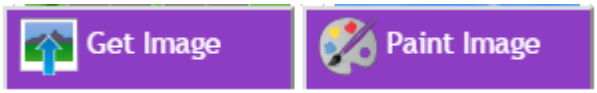
The left-hand side of the screen contains the available objects, this varies according to the level of 2Code.

These can be dragged onto the main part of the screen to include them in the program.

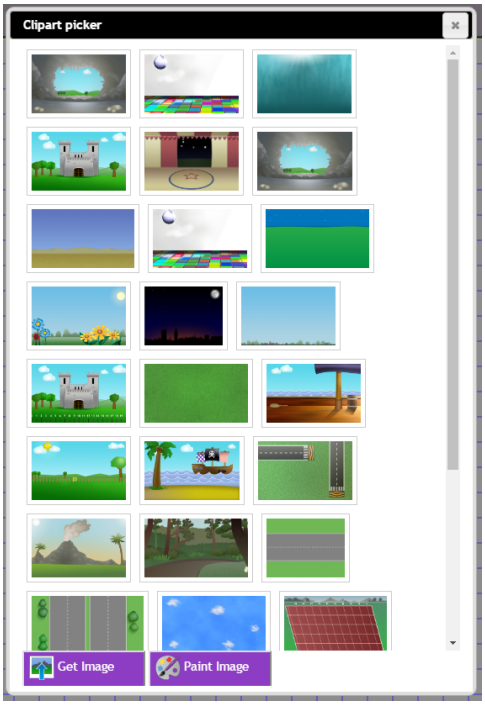
The background can be changed by clicking  .

Each object as well as the background has certain properties that can selected. The properties appear on the left-hand side below the object types when an object is added and clicked on. These are the properties of the background.

Property	Value
type	background
name	background
colour	
image	?
Grid size	4

The name, colour, image and grid size can be altered here. Any image can be used for a background image, there are some examples included in the clip art picker and you can also click the 

buttons to add your own.



The properties of each object type are discussed in more detail in the Chimp, Gibbon and Gorilla level sections.

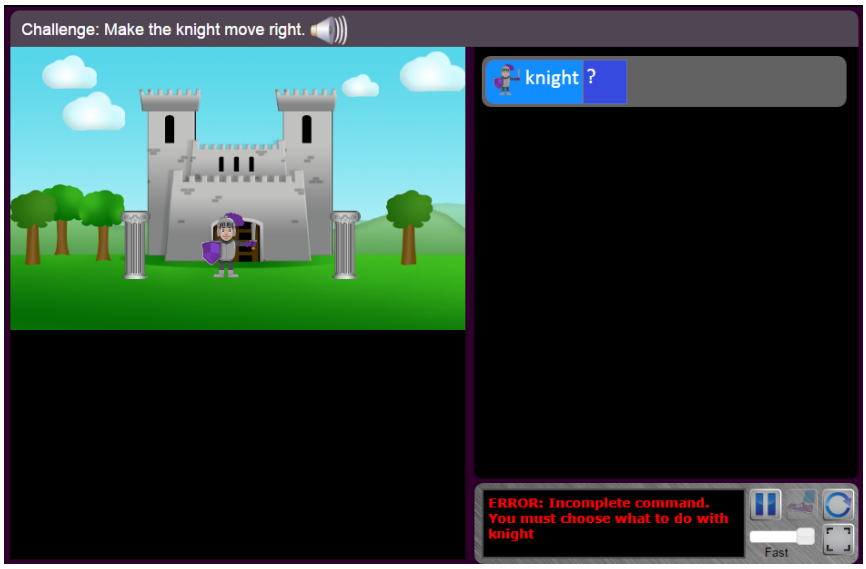
To exit the design view and return to the code view click 

.

2.1.3 Debug Tools - bottom right

The debug tools will help you to get a deeper understanding of what your computer is doing and also work out why things aren't working as expected.

The debug window appears when you are in play mode and is at the bottom right of the screen.



The console helps you understand what the program is doing. In the above example the log is red and it explains that the knight has no command to tell it what to do.

You can click on the lines of the log to see which piece of code caused each step.

 Use the pause button to run in pause mode; clicking the step button  will play one line of code at a time.

 Then click play to continue properly

When the code is run in full screen mode  you do not see these tools; when you share the code, it will open in full screen mode automatically.

 The restart button restarts the code again. You can click the pause button prior to restarting the code and then step through the code from the beginning; this is useful when demonstrating to the class.

2.1.4 Step mode

Clicking the  prior to pressing Play will set the code to run in step mode.

A line of code will run and then the code will pause until the Play button in the debug window is clicked. See the section Debug Tools for further information about this function.

2.1.5 A note about Objects in 2Code

You will notice as you move through the levels of 2Code, a progression in the way that objects behave and how they are named. This is deliberate but sometimes causes confusion.

In Chimp and Gibbon levels, there is an object called 'Food'. At Gorilla level, this object is renamed 'Object'. This is to highlight an important concept as children progress from Chimp through Gibbon to Gorilla: everything you add is an Object. The Button is an object, so too is the Number and Input, Text, Shape, Turtle, Character, Animal, and Vehicle. In the guided lessons, the custom object types, such as the Tuna, Trout and Clown objects, are subtypes of the Animal object; the Knight object in Guard the Castle is a subtype of the Character object. Introducing distinct types and then progressing to object once they're familiar with that is an introduction to **object-oriented thinking.**

The various objects have differing properties throughout the levels and these are detailed in the sections on each level. This is particularly noticeable in the Gorilla level.

For example, in the Gorilla level, the Animal object doesn't have direction because it has angle, which is a different form of movement to the Character Object and why we don't have one button for both types of object.

Similarly the Vehicle object doesn't have direction but does have angle.

We don't have one button for the Animal and Vehicle objects because they have a different default rotation style: the Vehicle faces the angle it's rotated to – whereas the Animal looks the same no

matter which way it is rotated.

This creates an interesting variety of movement styles. Try animating your Character and your Animal up a slope – the concept of “up”, “down”, “left” and “right” (Character) don’t work for this but rotating the angle while facing the same way (Animal) does. What if you want your character to move uphill? Create an Animal object and change its image to that of a character. There’s the interchangeability of the object types again, leading up to the abandonment of the term “Food” in Gorilla.

2.2 Chimp Level

Chimp level is the simplest level of 2Code. It is aimed at beginning coders. If you follow the guided lessons or scheme of work children will use this level in years 1 to 4 depending upon ability.

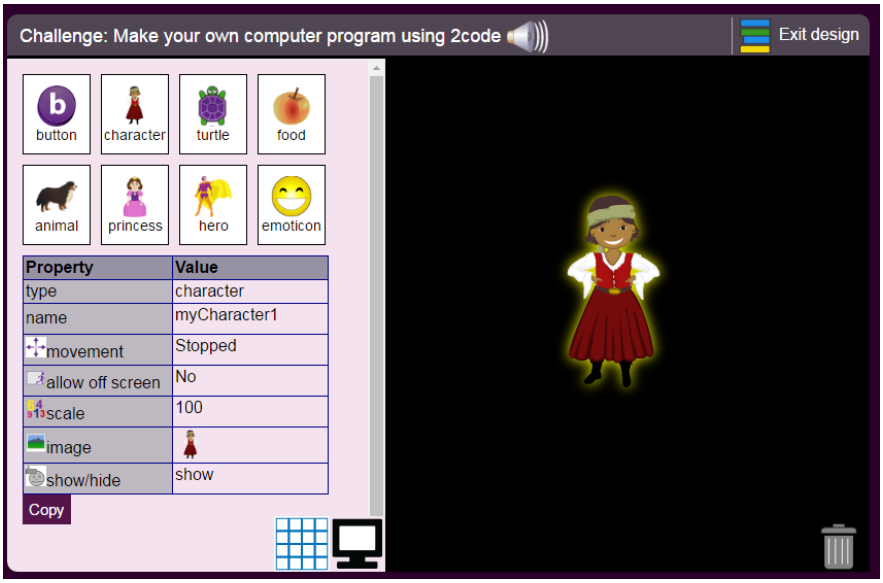
The following section explains the Objects and Commands that are available in Chimp level.

2.2.1 Objects

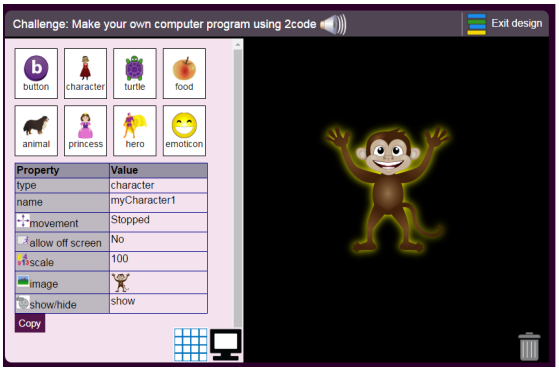
A 2Code program is created by adding objects such as characters in design view and then changing the properties and adding commands for them to follow.

The following is an example of this process and applies to all levels of 2Code.

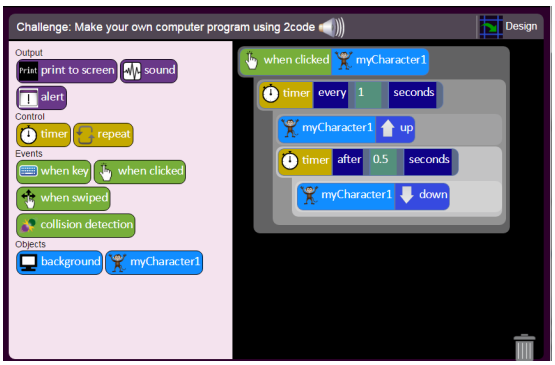
In design view, add a character by dragging it onto the coding window.



Set its image property to that of a monkey by double clicking on the image in the property box to the left.

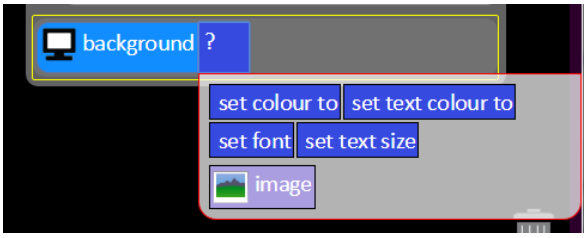


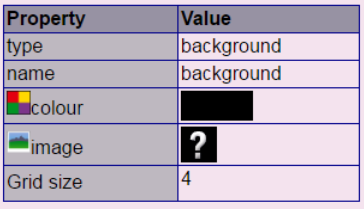
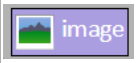
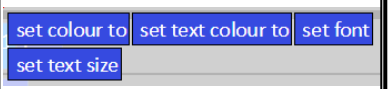
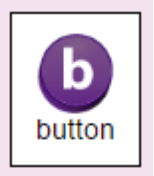
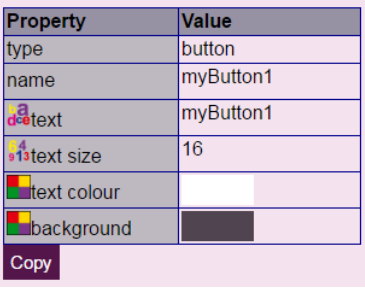
Exit design view and in code view, use commands to make the monkey look like it is jumping around when it is clicked.

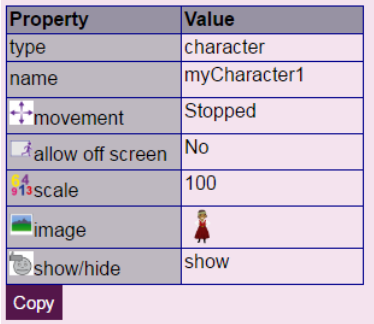
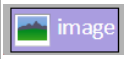
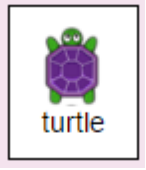
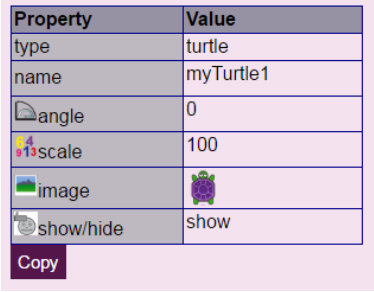
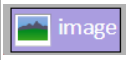
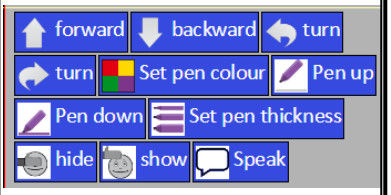


The table below shows the objects that are available in Chimp level. It also shows which properties and actions can be changed in design view, when you initially set up your design, or in code view. This means, for example, that you can change what an object looks like during the code running so you could turn a prince into a frog, or a daytime background into a night time background. The more advanced levels of 2Code allow more things to be changed.

When coding, the properties are shown in lilac and the actions in blue.



Object	Properties in Design View	Properties in Code View	Actions in code view
			 These set the background colour and also the properties of any text that is printed to the screen.
		None	None

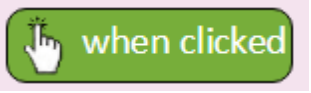
Object	Properties in Design View	Properties in Code View	Actions in code view
			 These make the object move in different directions. Stop, hide or show the object. Make the object speak by displaying a speech bubble.
			 A turtle moves in a similar way to a floor turtle using Logo type actions. Turn is by a number of degrees.

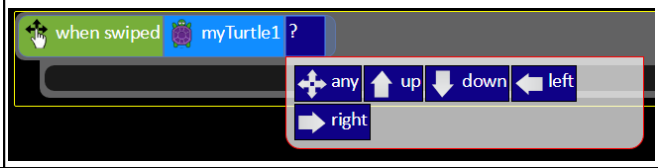
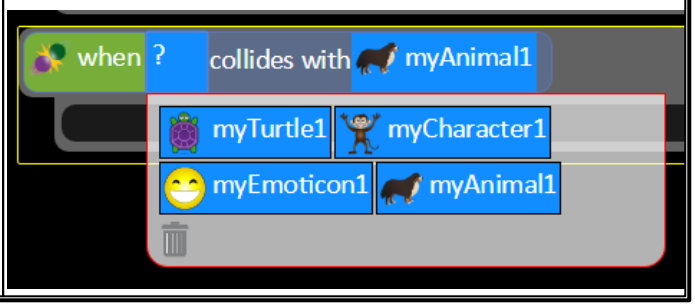
2.2.2 Commands

The different types of commands are colour-coded to indicate their functions. In Chimp level, this includes Output (purple), Control (yellow), Events (light green). A colour key is available for more details.

The commands available for each level are shown on the left-hand side above the objects.

When commands are dragged into the code window, 2Code will indicate the type of instructions that you need to add. For example, if you drag in a Print to Screen command, a text box will pop-up to indicate that you need to type in some text that the code will print to the screen. If you drag in a sound command, the sound picker window will open.

Command	Explanation
	Prints some text specified by the coder to the screen.
	Causes a sound to play. the sound picker will open for the coder to select a sound, when this code block is added to the code window.
	Creates a pop-up window with a message for the user and an OK button to click.
	Create a timer. The coder can select whether this time should run after a certain length of time or every x length of time. The time length is measured in seconds or quarter seconds. 
	Repeats the code inside it either forever or every x (or quarter seconds). 
	Runs the code inside it when the specified key is pressed. The code chooses which key (including arrow keys and space bar)
	Runs the code inside it when the object is clicked. The coder is given a choice of all the available objects.

Command	Explanation
when swiped	Useful for tablets. This command runs the code inside it when an object is swiped. The coder chooses the object and the direction of the swipe. 
collision detection	Runs the code inside it when two objects collide. The coder selects the two objects. 

2.3 Gibbon Level

Gibbon level is the second level of 2Code. If you follow the guided lessons or scheme of work children will use this level in years 4 to 6 depending upon ability.

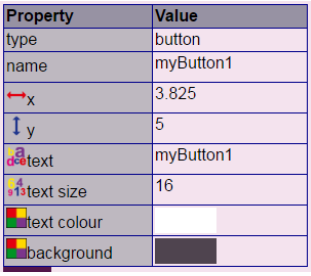
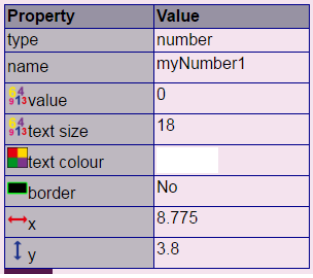
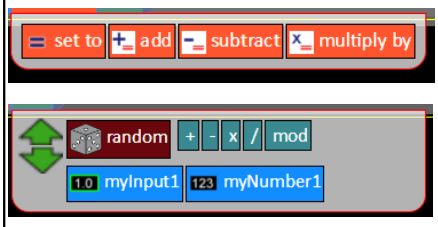
The following sections explain the additional Objects, Commands and functionality that are available in Gibbon level, **that are additional or replace, those in Chimp level**.

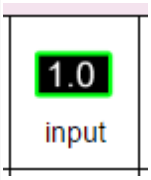
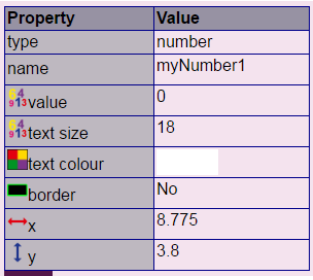
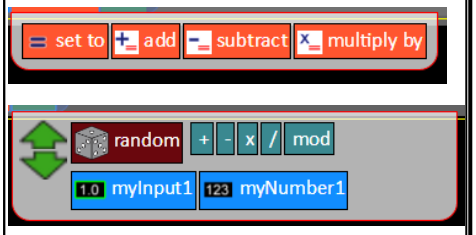
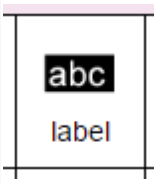
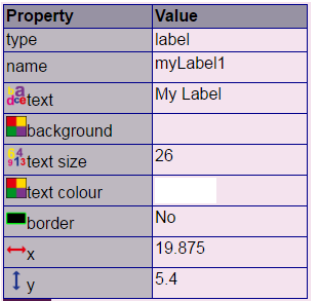
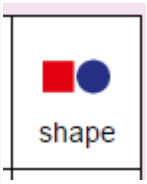
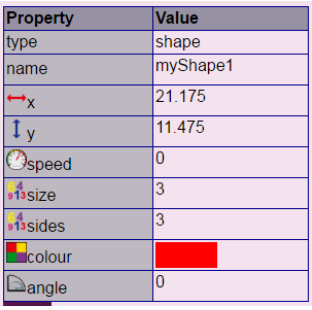
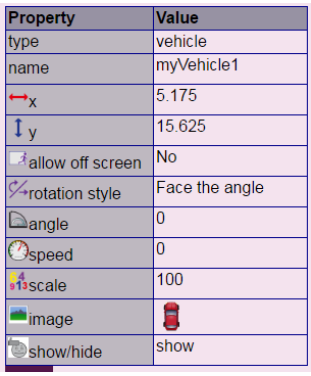
2.3.1 Gibbon Objects

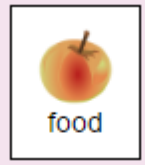
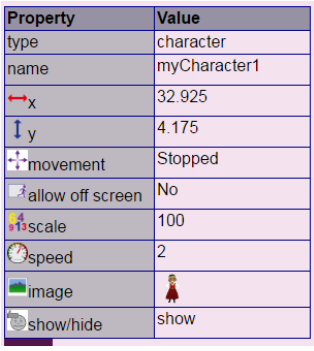
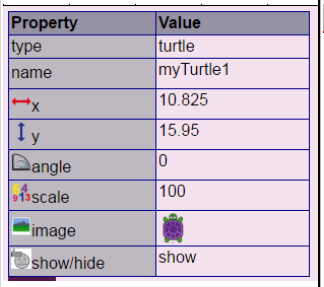
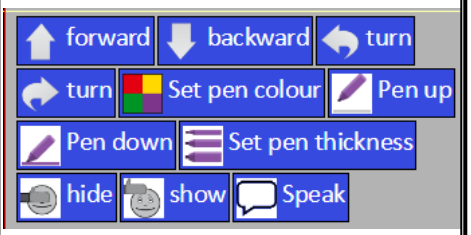
Some of the objects from the Chimp level now have additional properties in design mode such as the x- and y-coordinates.

Some of the Chimp object types do not appear in Gibbon as the user will now be familiar with changing the image of an object such as a character to make it look like a hero or princess if they wish.

Some of the object types are new.

Object	Properties in Design View	Properties in Code View	Actions in code view
		None	None
			 **This displays a number on the screen which can be set to different values and/or have calculations performed on it. In the image above, myInput1 and myNumber1 are objects that also have a number value and the value of these can be set to affect the value of the number object.**

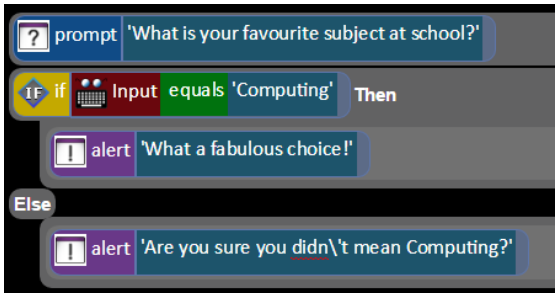
Object	Properties in Design View	Properties in Code View	Actions in code view
			 **This displays an input box on the screen into which a number can by typed.** **In the code, input can be set to different values and/or have calculations performed on it.**
		None	None. The text for the label is set in design view and cannot be changed.
		 The options offered will depend upon the property selected.	
		 The options offered will depend upon the property selected.	

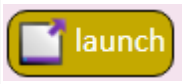
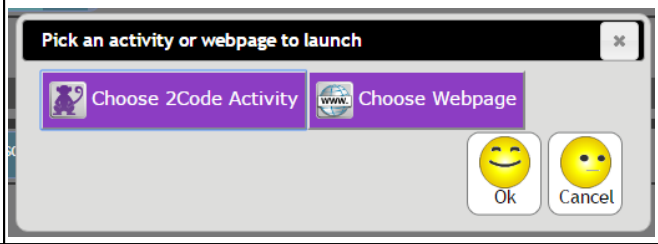
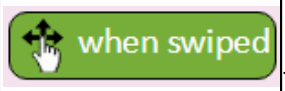
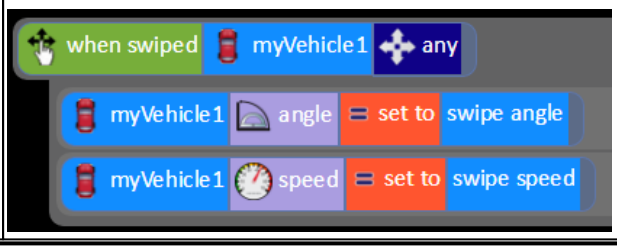
Object	Properties in Design View	Properties in Code View	Actions in code view
  			 These make the object move in different directions. Stop, hide or show the object. Make the object speak by displaying a speech bubble.
			 A turtle moves in a similar way to a floor turtle using Logo type actions. Turn is by a number of degrees.

2.3.2 Gibbon Commands

The Gibbon level introduces Input commands, shown dark blue. There are also additional Control commands including selection.

Variables are also introduced in the Gibbon level. These are explained in the Variables section.

Command	Explanation
get input	This command will put a cursor in the top left of the screen and get the input typed onto the screen. For example if you have an alert that asks the user to type their name, you can use this to print their name back to them: 
prompt for input	This combines the alert and get input functions, a pop-up screen will ask the user to enter something and they type it into a text box on the pop-up screen.
	This runs the code inside if a certain condition is met. The condition could depend upon something entered by the user or upon the value of a variable.
if/else	This runs the code inside the first block if a certain condition is met, otherwise it runs the code inside the second block. 
repeat until	This command repeats the code inside until a certain condition is met. The condition could depend upon something entered by the user or upon the value of a variable.
restart	This will restart the program from the beginning. Useful if you want to include a restart button in your program.

Command	Explanation
	This will launch another 2Code program or open a web page. The following screen will allow the coder to select which. You might want buttons in your program to link to other programs that you have made or to the Internet. The launch command can be useful when you write much bigger programs as you can split them into smaller chunks that launch each other. 
	In Gibbon level, you are now able to use the swipe speed and swipe angle in the code. This works especially well when applied to objects that can have both their angle and speed set, such as vehicles. Remember that you can change the image of a vehicle to anything else such as a person if you want to be able to do this with objects that don't look like vehicles. 

2.3.3 Variables

Variables are used in a program to remember a piece of information.

In 2Code this can be text or a number.

It is usual to create all variables at the beginning of the code.

To create a variable, drag the 

command into the code window. You will then be offered the choice of a number or text variable.

You should then give the variable a meaningful name so that you can easily add to or debug your program and you do not get confused about which variable is which.

You can set the initial value of the variable by typing it in and you are also given the choice to set it to a random animal, noun, verb or adjective.

To change the value of the variable use the 

command.

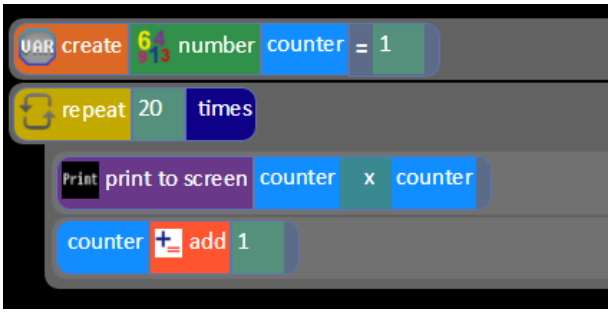
When you drag this into the code window, first select the variable by clicking on its name, then 

set to or 

. If the variable is a text variable, the add command will concatenate the text. If it is a number it will add the new number on.

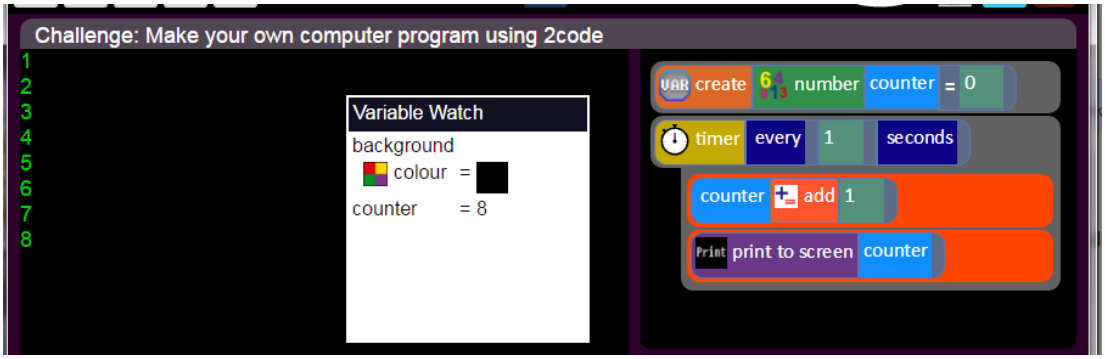
The following code will produce a list of random sentences:

This code will produce the first 20 square numbers:



You can save input from the user into a variable and use this:

When in Play mode, the variable watch window will show the value of all of the variables for every object. The example below shows the variable watch window, the code and the output for a counter program.



2.4 Gorilla Level

Gorilla level is the third and highest level of 2Code. If you follow the guided lessons or scheme of work children will use this level in years 4 to 6 depending upon ability.

The following section explains the additional Objects, Commands and functionality that are available in Gorilla level **that are additional or replace, those in Gibbon level**.

2.4.1 Gorilla Objects

The text object replaces the Label object. This has more properties to define the look of the text and can be hidden/shown.

The vehicle, animal and character objects have an additional property called friction. This makes the object slow down after it has started moving to model friction in real-life.

There is a new object called Walls that appears in design view next to the background object. You use this to draw barriers in your program and events can occur when other objects collide with the walls.

The food object has been renamed 'Object' and highlights an important concept as children progress from Chimp through Gibbon to Gorilla: everything you add is an Object. The Button is an object, so too is the Number and Input, Text, Shape, Turtle, Character, Animal, and Vehicle. In the guided lessons, the custom object types, such as the Tuna, Trout and Clown objects, are subtypes of the Animal object; the Knight object in Guard the Castle is a subtype of the Character object. Introducing distinct types and then progressing to object once they're familiar with that is an introduction to object-oriented thinking.

The various object have differing properties throughout the levels and these are detailed in the sections on each level. This is particularly noticeable in the Gorilla level.

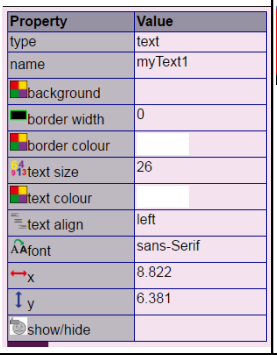
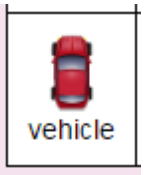
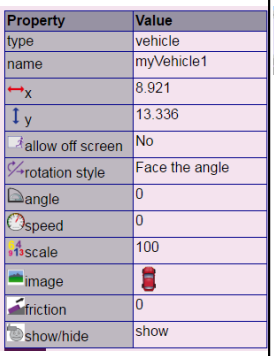
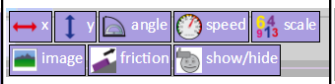
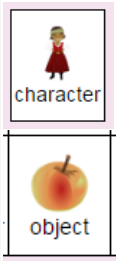
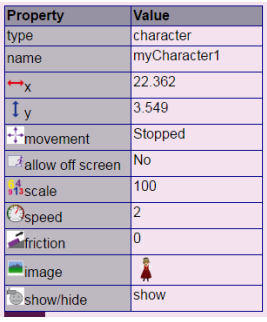
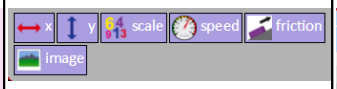
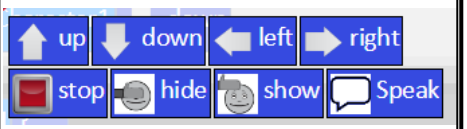
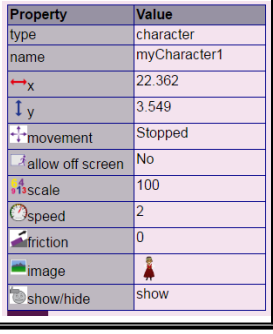
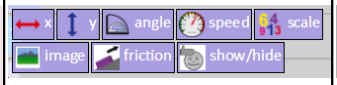
For example, in the Gorilla level, the Animal object doesn't have direction because it has angle, which is a different form of movement to the Character Object and why we don't have one button for both types of object.

Similarly the Vehicle object doesn't have direction but does have angle.

We don't have one button for the Animal and Vehicle objects because they have a different default rotation style: the Vehicle faces the angle it's rotated to – whereas the Animal looks the same no matter which way it is rotated.

This creates an interesting variety of movement styles. Try animating your Character and your Animal up a slope – the concept of "up", "down", "left" and "right" (Character) don't work for this but rotating the angle while facing the same way (Animal) does. What if you want your character to move uphill?

Create an Animal object and change its image to that of a character. There's the interchangeability of the object types again, leading up to the abandonment of the term "Food" in Gorilla.

Object	Properties in Design View	Properties in Code View	Actions in code view
			Show/Hide
			
			
			

2.4.2 Using tabs

Tabs are introduced in the Gorilla level because children are probably writing increasingly complex programs. These help them organise their program into different tabs.

To add a tab, click the plus symbol 

This will require you to give your tabs names. Name them with something that indicates their function so that code can easily be found.

2Code runs the code from left to right. To see this in action, run the code in step mode.

 There is a helpful video about the use of tabs in the video help area of 2Code. This shows how to move code between tabs and how to delete and reorder tabs.

 To move code, select the block that you wish to move then click on the purple arrow which appears next to the tabs. Select the tab to move the code to.

To re-order tabs, drag them into the order that you wish them to have.

Can move code to a different tab using the arrow key

To delete tab, drag it to the bin at the bottom right.

In run mode, the tabs switch view to wherever the currently active code is.

2.5 Glossary

A searchable glossary of coding terms used within 2Code can be found on the main 2Code page as well as on the 2Code teacher page

2.6 2Code Colour Key

This is a summary of what the different colours used within 2Code denote.

For more information on these terms please refer to the rest of this guide, to the code glossary and the tutorial videos.

Colour	Meaning	Description
	Output command	A command that generates output from the computer such as a sound from the speakers or something appearing on the screen.
	Input command	Commands that ask the user for input.
	Control command	Commands that control the programme such as timers, if/else, repeat or restart commands.
	Events	Events are blocks of code that are run when something happens such as a key press or a mouse click.
	Create or change variable command	Commands that create or change variables. For information on variables check out the tutorial videos and the glossary.
	Variable or object	Used for storing pieces of information within a programme. Objects are variables that can be created in design mode.
	Variable type	The type of variable. In 2Code variables can either be created as text or a number. An object is also a type of variable but in 2Code objects are created in design mode.
	Action	A command run on an object to alter its behaviour, such as "UP" or "DOWN"
	Property	A variable associated with an object, such "x", "y" or "scale".
	Number	A number.
	Text	A piece of text.
	Assignment operator	A type of operator that is used to assign or reassign (or change) the value of a variable.
	Mathematical operator	An operator which functions as a typical mathematical statement.
	Conditional operator	An operator (symbol) which evaluates to either true or false depending on the values either side of it.
	Logical Operator	Logical operators are used for combining conditions, allowing for complex tests to be created. The most common examples of logical operators are "AND" and "OR".
	Sound	A sound.

2.7 Sharing

Once a 2Code program has been saved into Purple Mash, it can be shared by clicking on the globe button in the toolbar  .

See the sharing guide in Purple Mash for further information about sharing work.

Links to your file will launch the 2Code program in a web browser in full screen mode and the embed code can be used to embed a 2Code program into a blog or a website. A Purple Mash login is not required to run a shared 2Code program.

2.8 Real code mode

On the more advanced lessons and Free Code modes it is possible to click the real code button and see the code in a simple subset of JavaScript.

The code can be edited in “real code” mode and clicking the “edit blocks” button will bring the user back to the usual graphical representation.

If the user types code into real code window that is syntactically incorrect (e.g. deleting a required } curly bracket), the real code window will flash red. Changing back to “edit blocks” will restore the code to the last state that was syntactically valid.

3 Guided Lessons

2Code contains a series of lessons which will guide children through creating simple programs. The lessons are split into stages; Chimp, Gibbon and Gorilla. See the table below for suggested age ranges for these.

The lessons are found on the main 2Code page. There are also guided assessment tasks at the end of each level, these test whether children have grasped the coding skills of that level and produce a report for teachers of how well the children tackled the challenges. The assessment tasks are not visible to pupils unless set as 2Dos so children cannot try them out unless they are set for them.

Use these links for

Solutions

Lesson objectives

Guided Assessment tasks.

Additional assessment materials

All these resources can be found in Purple Mash Teachers area/2Code Teachers area/Guided Lessons & Solutions.

There are three levels of lessons. Approximate years are given but in practice the year for particular children will vary.

Name	Purpose	Approximate Year
Chimp	Covers basic coding/algorithms and debugging.	1-4
Gibbon	Introduce more complex ideas such as variables and selection.	4-6
Gorilla	More advanced lessons that will guide the child into creating games or quizzes.	4-6

Linked to these, are the Debugging challenges, which are discussed in the relevant section.

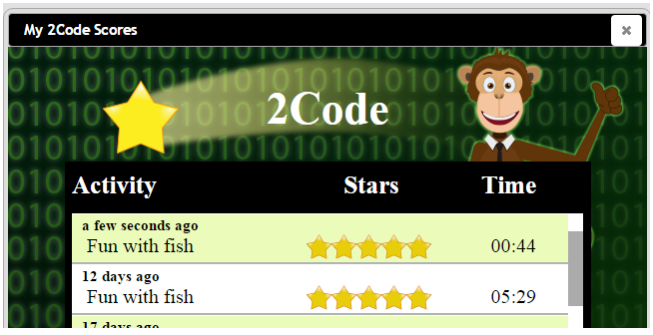
3.1 Scores

In a 2Code activity pupils are awarded 5 stars if they don't use any hints. Each hint reduces the number of stars they are awarded by 1. These scores are reported to teachers in the same way as other Purple Mash activities by using the Scores Report tool in the Teachers Admin section.

	A	B	C	D	E	F	G	H	I	J
1	Pupil Statistics Export - Run On 20-03-2017 at 12:06									
2										
3	Applications:	All								
4	Run For:	Ladybirds (Class)								
5	For The Period:	All Stats								
6										
7			2Code (Bubbles)		2Code (2DIY - Catching)		2Code (Fireworks 2)		2Code (Fun with fish)	
8			Latest Score	Oldest Score	Latest Score	Oldest Score	Latest Score	Oldest Score	Latest Score	Oldest Score
18	Chris Kos	Ladybirds	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	3 Stars Time:50	3 Stars Time:50	2 Stars Time:50	2 Stars Time:50
19	Casey Dein	Ladybirds	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	3 Stars Time:50	3 Stars Time:50	3 Stars Time:50	3 Stars Time:50
20	Mia Ritu	Ladybirds	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50
21	Dane Harley	Ladybirds	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	2 Stars Time:50	2 Stars Time:50	1 Stars Time:50	2 Stars Time:50
22	Delilah Singh	Ladybirds	5 Stars Time:50	5 Stars Time:50	1 Stars Time:50	5 Stars Time:50	3 Stars Time:50	3 Stars Time:50	2 Stars Time:50	3 Stars Time:50
23	Doll Reddin	Ladybirds	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	3 Stars Time:50	3 Stars Time:50	2 Stars Time:50	2 Stars Time:50
24	Elena Diva	Ladybirds	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	4 Stars Time:50	5 Stars Time:50	4 Stars Time:50	5 Stars Time:50
25	Ethan Thomas	Ladybirds	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	5 Stars Time:50	2 Stars Time:50	3 Stars Time:50	1 Stars Time:50	1 Stars Time:50
26	Ethan Wedgewood	Ladybirds	5 Stars Time:50	5 Stars Time:50	3 Stars Time:50	3 Stars Time:50	2 Stars Time:50	2 Stars Time:50	1 Stars Time:50	1 Stars Time:50
27	Summary		Average score : 5 stars	Average score : 5 stars	Average score : 4 stars	Average score : 5 stars	Average score : 3 stars	Average score : 3 stars	Average score : 2 stars	Average score : 3 stars
28										
29										
30	Legend									
31	Red	1 Star								
32	Orange	2 Stars								
33	Yellow	3 Stars								
34	Light green	4 Stars								
35	Dark green	5 Stars								
36	Comment	In a 2Code activity pupils are awarded 5 stars if they don't use any hints. Each hint reduces the number of stars they are awarded by 1.								
37										

Pupils can see their scores for the guided lessons using the My 2Code Scores link on the main 2Code page.

This shows the activities that they have done, when they did the activity, the number of stars achieved and the time taken.



4 Free code

In Free Code mode, children can create any kind of program they like. There is a Free Code Tool for each of the levels of 2Code. The Free Code button is found on the main 2Code Page, in each of the levels sections and also in a section called Free Code.

The Purple Mash Computing Scheme of Work has a Coding Unit in each year group that uses Free Code as well as some of the guided activities. Details of the Scheme of Work can be found on the Scheme of Work pages or in the Free Code Plans and Resources section of the teacher 2Code area.

This table contains a summary of the **Coding Unit** contents in each year group of the Scheme of Work. The lessons show a progression of knowledge and skills from lesson to lesson and year to year. Children who have not used 2Code before will benefit by starting on lessons aimed at younger children. Teachers who are not familiar with the tools in 2Code might find reviewing the lessons for younger children helpful to build up their own knowledge.

Year	Lesson 1	Lesson 2	Lesson 3	Lesson 4	Lesson 5	Lesson 6
1	Introduction to coding	Block coding on screen	Backgrounds and characters	Making characters move	'Clicked on' command	Collision detection
2	Algorithms	Repeat and Timer	Debugging	Logical reasoning	Retelling a story in code	
3	Accomplishing a goal in a program	Simulating a physical system	Repetition using a timer	'If' statements	Debugging	Variables
4	Accomplishing a goal in a program	Variables and 'if/else' statements	Repetition and user input	Debugging	Variables	Using 2Code to make a control simulation
5	Accomplishing a goal in a program	Simulating a physical system	Introducing text variables	Creating a game with a score and timer		Internet safety
6	Designing and writing a more complex program		Introducing Functions	Vocabulary review and quizzes	Using buttons to showcase work	Using 2Code to make a text based adventure

The tables on the following pages show the lesson aims and success criteria in each year group.

Year 1 Year 2 Year 3
Year 4 Year 5 Year 6

4.1 Year 1

Lesson	Aims	Success Criteria
1	Introduction to coding.	• Children can explain what is meant by coding.
2	Introduction to block coding on screen.	• Children can explain what a block of code is. • Children can read through combined blocks of code. • Children know that for the computer to make something happen, it needs to follow clear instructions.
3	Introduction to backgrounds and characters.	• Children can use Design Mode to have control over how my game looks.
4	Making a character move left and right.	• Children can write a program that controls how a character moves. • Children can explain what is happening and write down/ talk through my code.
5	Making a character move when clicked.	• Children can write a program that controls how a character moves and stops when clicked.
6	Introduction to Collision Detection.	• Children can write a program where objects can stop moving and a sound is played when the objects collide.

4.2 Year 2

Lesson	Aims	Success Criteria
1	To introduce algorithms.	• Children can explain that an algorithm is a set of instructions. • Children can explain that for the computer to make something happen, it needs to follow clear instructions. • Children can show their computer program and point out the algorithms they created.
2	To use Repeat and Timer commands.	• Children can explain how to use the following terms in a computer program: Command, Repeat, Input, Output, Event, Collision Detection and Timer. • Children can create a computer program including at least four of the above new coding vocabulary terms.
3	Debugging.	• Children can explain what debug (debugging) means. • Children can explain what they did so that their computer program did not work. • Children can debug simple programs.
4	To explore the possible actions of different types of objects.	• Children can create a computer program using different objects. • Children can predict what the objects in classmates' programs will do, based on my knowledge of the objects' limitations, e.g. a turtle can only move in specific ways. • Children can explain how they know that certain objects can only move in certain ways.
5	To create a more complex program to retell a story, using 2Code.	• Children can plan and use algorithms in programs successfully to achieve an end result. • Children can code a program using a variety of objects, actions, events and outputs successfully.

4.3 Year 3

Lesson	Aims	Success Criteria
1	To design and write a program that accomplishes a specific goal.	• Children can explain what Object, Action, Output, Control and Event are in computer programming. • Children can explain which commands they included in their program and what they achieve.
2	To design and write a program that simulates a physical system.	• Children can explain how their program simulates a physical system, i.e. my vehicles move at different speeds and angles. • Children can describe what they did to make their vehicle change angle. • Children can show that their vehicles move at different speeds.
3	To use repetition commands.	• Children can show how their character repeats an action and explain how they caused it to do so. • Children are beginning to understand how the use of the timer differs from the repeat command and can experiment with the different methods of repeating blocks of code. • Children can explain how they made objects repeat actions.
4	To introduce 'if' statements.	• Children can create an 'if' statement in their program. • Children can use a timer and 'if' statement to respond to the actions of a character and change their actions.
5	Debugging.	• Children can explain what steps to follow to debug a program. • Children can explain what they did so that my computer program did not work. • Children can explain how they debugged a partner's program.
6	To introduce variables.	• Children can explain what a variable is in programming. • Children can explain why variables need to be named. • Children can create a variable in a program. • Children can set/change the variable values appropriately to create a timer.

4.4 Year 4

Lesson	Aims	Success Criteria
1	Design and write a program that accomplishes a specific goal.	• Children can explain what Object, Action, Output, Control and Event are in computer programming. • Children can explain which commands they included in their program and what they achieve.
2	Variables and 'If/else' statements.	• Children can create an 'If/else' statement. • Children understand what a variable is in programming. • Children can set/change the variable values appropriately.
3	Using repetition and user input.	• Children can show how a character repeats an action and explain how they caused it to do so. • Children can make a character respond to user keyboard input.
4	Debugging.	• Children can explain what steps I need to follow to debug a program. • Children can explain what they did so that their computer program would not work. • Children can explain how they debugged their partner's program.
5	Working with variables.	• Children can explain what a variable is when used in programming. • Children can create a timer that prints a new number to the screen every second. • Children can explain how they made their program change the number every second.
6	Using 2Code to make a control simulation	• Children can create an algorithm modelling the sequence of a simple event. • Children can manipulate graphics in the design view to achieve the desired look for the program. • Children can use an algorithm when making a simulation of an event on the computer.

4.5 Year 5

Lesson	Aims	Success Criteria
1	Designing and writing a program that accomplishes a specific goal.	• Children can explain what Object, Action, Output, Control and Event are in computer programming. • Children can explain which commands they included in their program and what they achieve.
2	Simulating a physical system.	• Children can explain how their program simulates a physical system, i.e. objects move at different speeds and angles. • Children can describe what they did to make their vehicle change angle. • Children can show that their vehicles move at different speeds.
3	Introducing text variables.	• Children can explain what a variable is in programming. • Children can set/change the variable values appropriately. • Children know some ways that text variables can be used in coding.
4 & 5	Creating and improving a game.	• Children can create a game which has a timer and score pad. • Children can use variables to control the objects in the game. • Children can create loops using the timer and If/else statements.
6	Internet safety.	• Children can explain what internet safety is. • Children can include two buttons that launch windows to two separate websites that provide further information in my program. • Children can use my coding knowledge to create a program that explains internet safety.

4.6 Year 6

Lesson	Aims	Success Criteria
1 &2	Designing and writing a more complex program that accomplishes a specific goal.	• Children can plan a program before coding to anticipate the variables that will be required to achieve the desired effect. • Children can follow through plans to create the program. • Children can debug when things do not run as expected.
3	Introducing functions.	• Children can explain what functions are and how they can be created and labelled in 2Code. • Children can explain how to move code from one tab to another in 2Code. • Children can explain how they organised code in a program into functions to make it easier to read.
4	Vocabulary review.	• Children are familiar with the vocabulary used throughout 2Code. • Children can describe coding using the appropriate terms.
5	Using buttons to showcase work.	• Children can include buttons that launch other programs, including their own. • Children can include buttons that launch windows to external websites.
6	Using 2Code to make a text based adventure	• Children can follow through the code of how a text adventure can be programmed in 2Code. • Children can adapt an existing text adventure to make it unique to my requirements.

5 Debugging Challenges

A debugging challenge consists of a broken program that the child is prompted to fix.

There are debugging challenges for each level of 2Code and these are incorporated into both the Free Code lessons and the Guided lessons.

The debugging challenges are found on the main 2Code page beneath the guided lessons.

6 Coding Principles

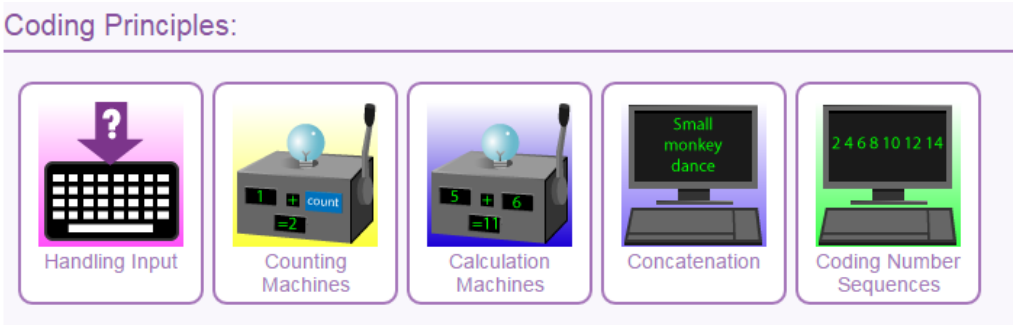
These are some additional activities that are referenced by some of the scheme of work lesson plans.

They explore coding functionality and coding structure that is common in many coding languages used to write programs that we use.

They are structured in the same way as the guided lessons and work as self contained activities to introduce children to a topic and give them practical tasks to complete.

They are separated from the general guided activities because they are activities without graphical embellishments that focus on the pure principals of coding – looping, handling input and arithmetic.

The current Coding Principles activities are:



7 Quizzes

Also on the main 2Code page are Vocabulary Quizzes. These test children's understanding of vocabulary and coding concepts.

There are 4 levels of increasing difficulty that can be set as 2dos for children. There are level quizzes which incorporate more than 1 level of quiz, children need to score full marks on a level before the quiz will move onto the next level.

You could use a lower level to refresh children's knowledge before beginning a unit of coding work and use a higher level afterwards.

Scores for these quizzes are reported in the same way as other 2DIY quizzes.

8 Games

Also on the main 2Code page is a Game section.

The games are split into the different levels of 2Code and test children's understanding of coding concepts.