



2Code User Guide

Contents

1. Introduction	4
2. Getting started.....	6
How 2Code is structured	6
Design view	8
Debug Tools - bottom right	10
Step mode	11
A note about Objects in 2Code	11
Chimp Level	13
Objects	13
Commands	16
Gibbon Level	18
Gibbon Objects	18
Gibbon Commands	21
Variables	23
Gorilla Level	25
Gorilla Objects	25
Using tabs	27
Glossary	27
2Code Colour Key	28
Sharing	29
Real code mode	29
3. Guided Lessons.....	30
Scores	31
4. Free code	32
5. Debugging Challenges.....	32
6. Coding Principles.....	32
7. Assessments.....	33
Skills Assessed by Level	34
Skills Covered - Chimp	34
Skills Covered - Gibbon	36
Skills Covered - Gorilla	37
Conducting the Assessments	38
Tasks and Solutions	39

Need more support? Contact us:



Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)

Chimp 1 - Purple Disco	39
Chimp 2 - Turtle's Day Out	43
Gibbon 1 - Shape Game	49
Gibbon 2 - Space Race	54
Gorilla 1 - Happy Eater Part 1	61
Gorilla 2 - Happy Eater Part 2	69
8. Quizzes	75
9. Games	75

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 and assessments** 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 split 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:

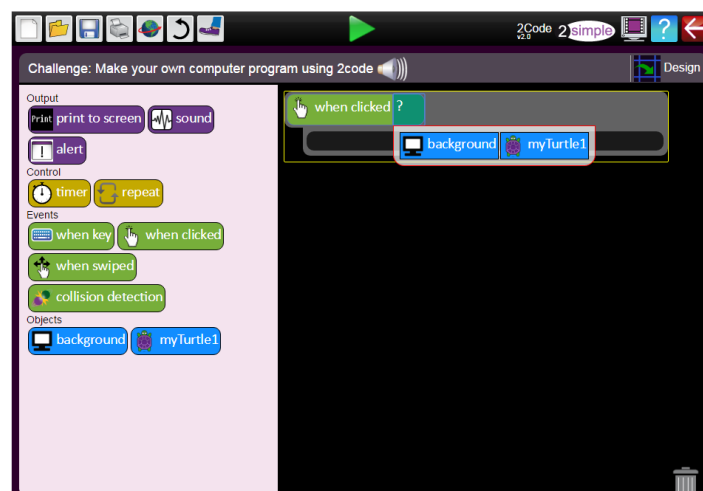
- [The glossary](#)
- [The colour key](#)
- [Sharing](#) 2Code programs
- [Real code mode](#)

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.



Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)

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

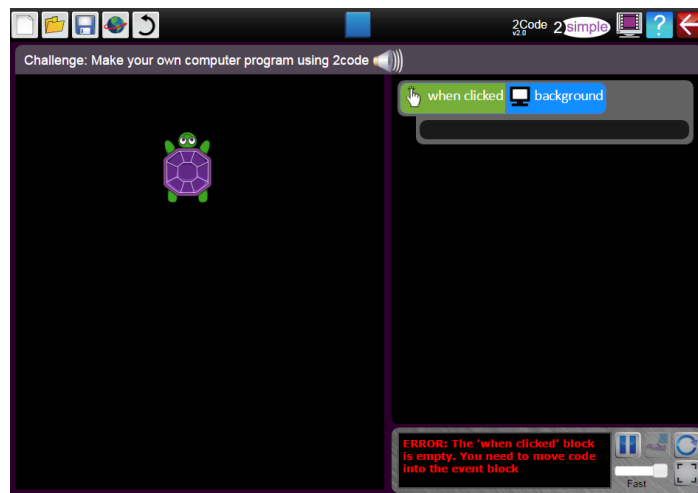
The main section in the middle is the main code window.

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.



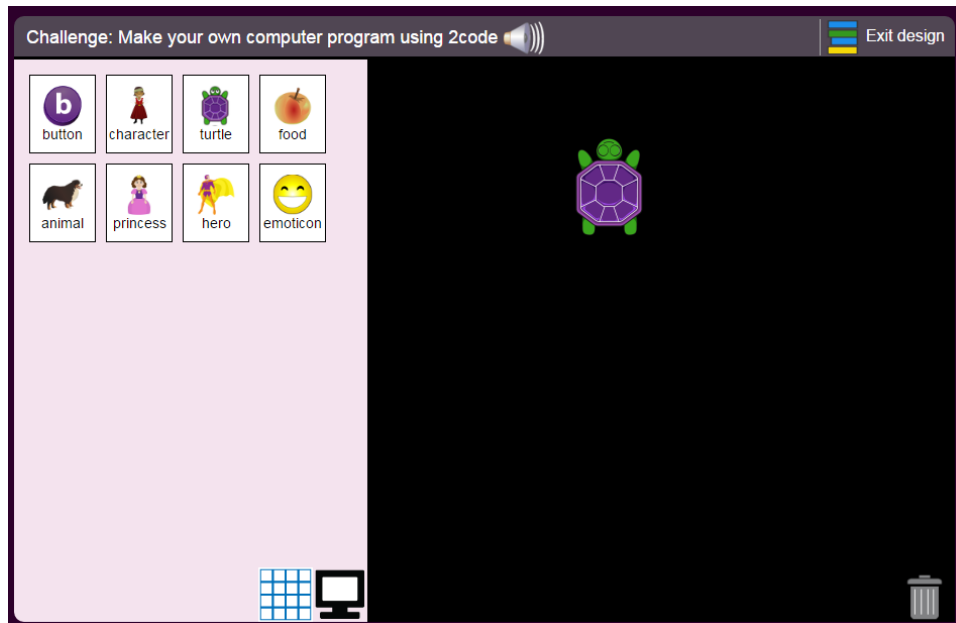
In Play mode, a debug window appears at the bottom right of the screen, this highlights errors and enables you to use the debug tools. See the [debug section](#) for further details of these.

2.1.1 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 be 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

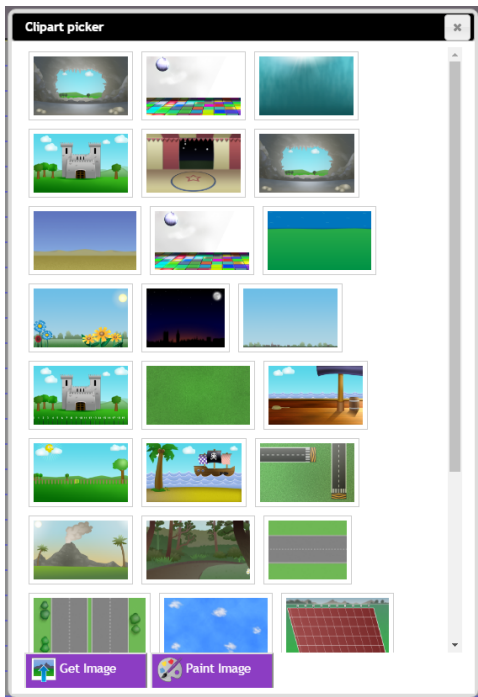
Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

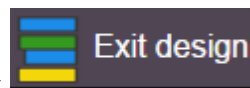
The name, colour, image and grid size can be altered here. Any image can be used for a background image, there are some examples included in the clip art picker and you can also click the



buttons to add your own.



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

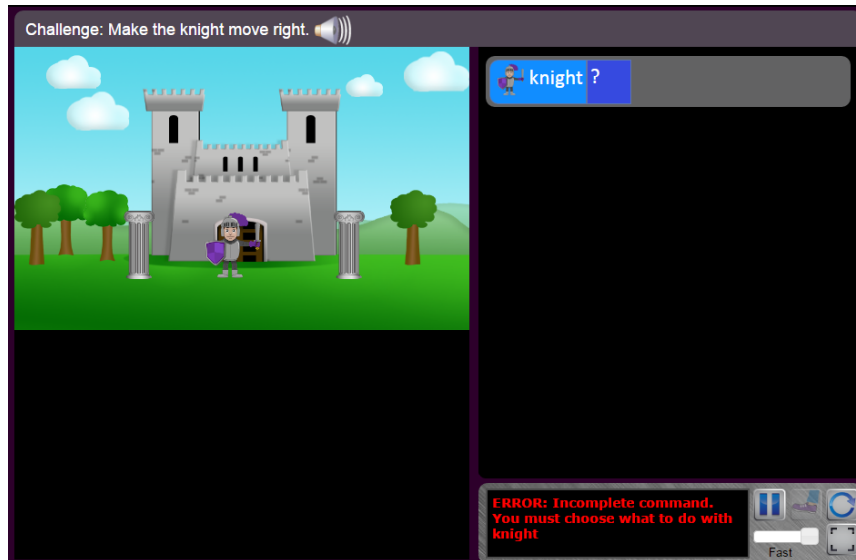


To exit the design view and return to the code view click

2.1.2 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.

2.1.3 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.

2.1.4 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.

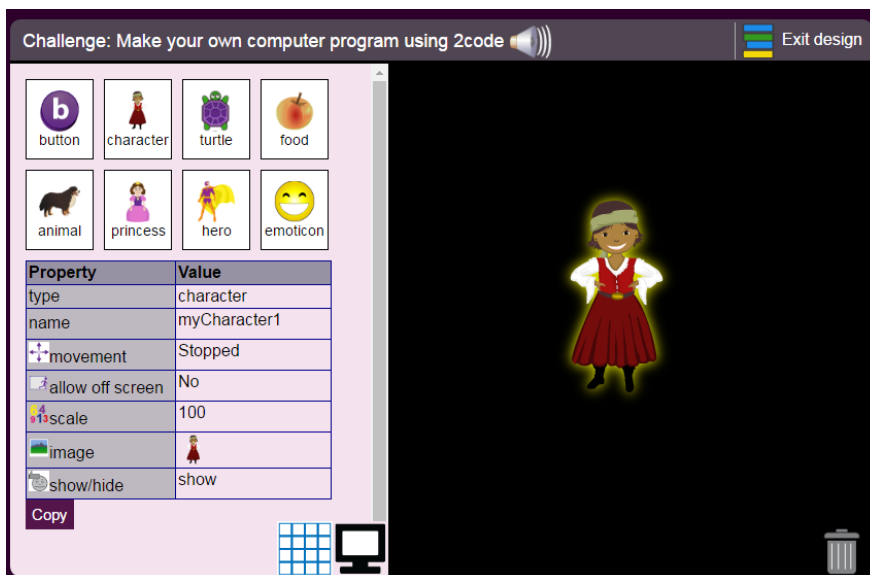
The following section explains the [Objects](#) and [Commands](#) that are available in Chimp level.

2.2.1 Objects

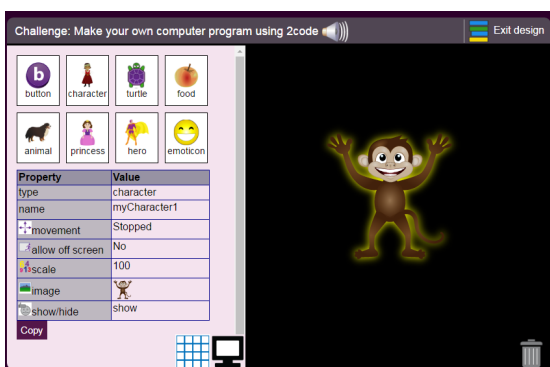
A 2Code program is created by adding objects such as characters in [design view](#) and then changing the properties and adding commands for them to follow.

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.

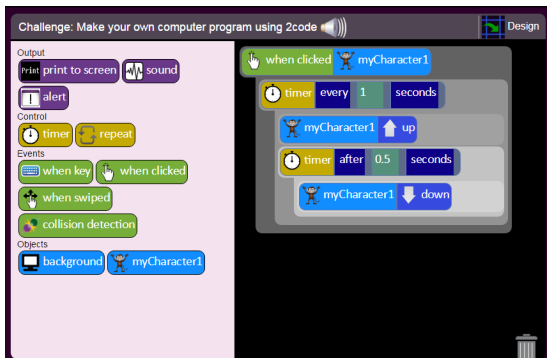


Exit design view and in code view, use commands to make the monkey look like it is jumping around

Need more support? Contact us:

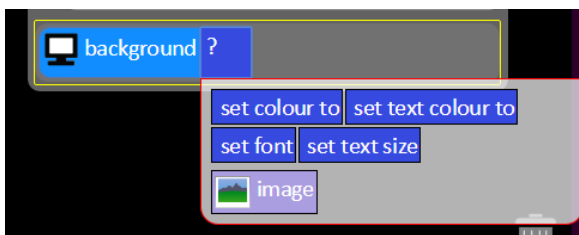
Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)


































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																
	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>background</td></tr><tr><td>name</td><td>background</td></tr><tr><td> colour</td><td></td></tr><tr><td> image</td><td></td></tr><tr><td>Grid size</td><td>4</td></tr></tbody></table>	Property	Value	type	background	name	background	 colour		 image		Grid size	4	 image	<div><div>set colour to</div><div>set text colour to</div><div>set font</div><div>set text size</div></div> <p>These set the background colour and also the properties of any text that is printed to the screen.</p>				
Property	Value																		
type	background																		
name	background																		
 colour																			
 image																			
Grid size	4																		
	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>button</td></tr><tr><td>name</td><td>myButton1</td></tr><tr><td> text</td><td>myButton1</td></tr><tr><td> text size</td><td>16</td></tr><tr><td> text colour</td><td></td></tr><tr><td> background</td><td></td></tr><tr><td>Copy</td><td></td></tr></tbody></table>	Property	Value	type	button	name	myButton1	 text	myButton1	 text size	16	 text colour		 background		Copy		None	None
Property	Value																		
type	button																		
name	myButton1																		
 text	myButton1																		
 text size	16																		
 text colour																			
 background																			
Copy																			

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware







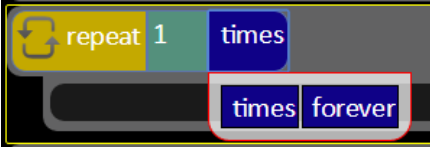
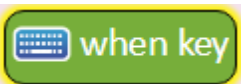
Object	Properties in Design View	Properties in Code View	Actions in code view																
<div><div> character</div><div> food</div><div> princess</div><div> emoticon</div><div> hero</div><div> animal</div></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>character</td></tr><tr><td>name</td><td>myCharacter1</td></tr><tr><td>movement</td><td>Stopped</td></tr><tr><td>allow off screen</td><td>No</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>image</td><td></td></tr><tr><td>show/hide</td><td>show</td></tr></tbody></table> <div>Copy</div>	Property	Value	type	character	name	myCharacter1	movement	Stopped	allow off screen	No	scale	100	image		show/hide	show	<div> image</div>	<div><div> up</div><div> down</div><div> left</div><div> right</div><div> stop</div><div> hide</div><div> show</div><div> Speak</div></div> <p>These make the object move in different directions.</p> <p>Stop, hide or show the object.</p> <p>Make the object speak by displaying a speech bubble.</p>
Property	Value																		
type	character																		
name	myCharacter1																		
movement	Stopped																		
allow off screen	No																		
scale	100																		
image																			
show/hide	show																		
<div><div> turtle</div></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>turtle</td></tr><tr><td>name</td><td>myTurtle1</td></tr><tr><td>angle</td><td>0</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>image</td><td></td></tr><tr><td>show/hide</td><td>show</td></tr></tbody></table> <div>Copy</div>	Property	Value	type	turtle	name	myTurtle1	angle	0	scale	100	image		show/hide	show	<div> image</div>	<div><div> forward</div><div> backward</div><div> turn</div><div> turn</div><div> Set pen colour</div><div> Pen up</div><div> Pen down</div><div> Set pen thickness</div><div> hide</div><div> show</div><div> Speak</div></div> <p>A turtle moves in a similar way to a floor turtle using Logo type actions. Turn is by a number of degrees.</p>		
Property	Value																		
type	turtle																		
name	myTurtle1																		
angle	0																		
scale	100																		
image																			
show/hide	show																		

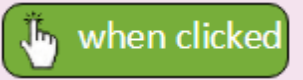
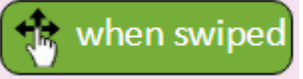
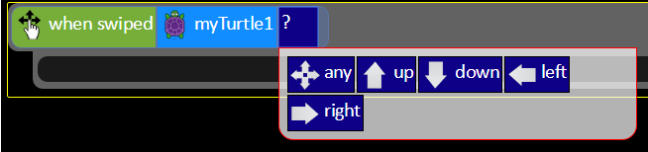

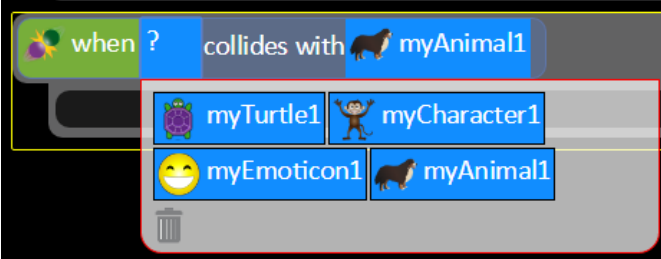
2.2.2 Commands

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.
	<p>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.</p> 
	<p>Repeats the code inside it either forever or every x (or quarter seconds).</p> 
	<p>Runs the code inside it when the specified key is pressed. The code chooses which key (including arrow keys and space bar)</p>

Command	Explanation
	<p>Runs the code inside it when the object is clicked. The coder is given a choice of all the available objects.</p>
	<p>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.</p> 
	<p>Runs the code inside it when two objects collide. The coder selects the two objects.</p> 

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.

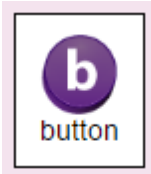

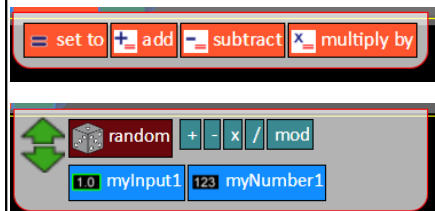
The following sections explain the additional Objects, Commands and functionality that are available in Gibbon level, **that are additional or replace, those in Chimp level.**

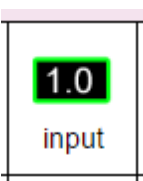
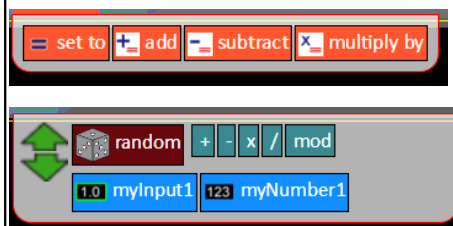




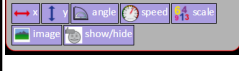
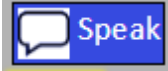
2.3.1 Gibbon Objects


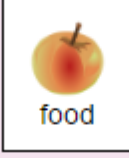
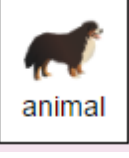


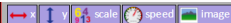





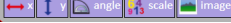


Some of the objects from the Chimp level now have additional properties in design mode such as the x- and y-coordinates.

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																		
	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>button</td></tr><tr><td>name</td><td>myButton1</td></tr><tr><td>x</td><td>3.825</td></tr><tr><td>y</td><td>5</td></tr><tr><td>text</td><td>myButton1</td></tr><tr><td>text size</td><td>16</td></tr><tr><td>text colour</td><td></td></tr><tr><td>background</td><td></td></tr></tbody></table>	Property	Value	type	button	name	myButton1	x	3.825	y	5	text	myButton1	text size	16	text colour		background		None	None
Property	Value																				
type	button																				
name	myButton1																				
x	3.825																				
y	5																				
text	myButton1																				
text size	16																				
text colour																					
background																					
	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>number</td></tr><tr><td>name</td><td>myNumber1</td></tr><tr><td>value</td><td>0</td></tr><tr><td>text size</td><td>18</td></tr><tr><td>text colour</td><td></td></tr><tr><td>border</td><td>No</td></tr><tr><td>x</td><td>8.775</td></tr><tr><td>y</td><td>3.8</td></tr></tbody></table>	Property	Value	type	number	name	myNumber1	value	0	text size	18	text colour		border	No	x	8.775	y	3.8		 <p>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.</p>
Property	Value																				
type	number																				
name	myNumber1																				
value	0																				
text size	18																				
text colour																					
border	No																				
x	8.775																				
y	3.8																				

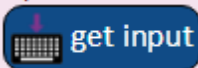




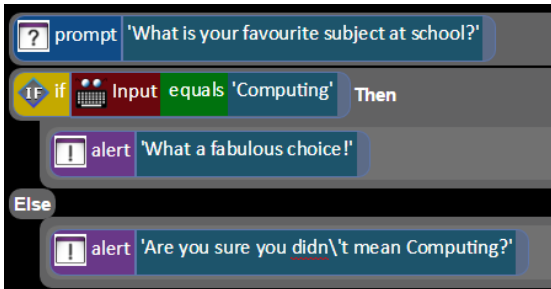
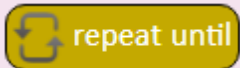

Object	Properties in Design View	Properties in Code View	Actions in code view																								
	<table><tr><th>Property</th><th>Value</th></tr><tr><td>type</td><td>number</td></tr><tr><td>name</td><td>myNumber1</td></tr><tr><td>value</td><td>0</td></tr><tr><td>text size</td><td>18</td></tr><tr><td>text colour</td><td></td></tr><tr><td>border</td><td>No</td></tr><tr><td>x</td><td>8.775</td></tr><tr><td>y</td><td>3.8</td></tr></table>	Property	Value	type	number	name	myNumber1	value	0	text size	18	text colour		border	No	x	8.775	y	3.8		 <p>This displays an input box on the screen into which a number can be typed.</p> <p>In the code, input can be set to different values and/or have calculations performed on it.</p>						
Property	Value																										
type	number																										
name	myNumber1																										
value	0																										
text size	18																										
text colour																											
border	No																										
x	8.775																										
y	3.8																										
	<table><tr><th>Property</th><th>Value</th></tr><tr><td>type</td><td>label</td></tr><tr><td>name</td><td>myLabel1</td></tr><tr><td>text</td><td>My Label</td></tr><tr><td>background</td><td></td></tr><tr><td>text size</td><td>26</td></tr><tr><td>text colour</td><td></td></tr><tr><td>border</td><td>No</td></tr><tr><td>x</td><td>19.875</td></tr><tr><td>y</td><td>5.4</td></tr></table>	Property	Value	type	label	name	myLabel1	text	My Label	background		text size	26	text colour		border	No	x	19.875	y	5.4	None	None. The text for the label is set in design view and cannot be changed.				
Property	Value																										
type	label																										
name	myLabel1																										
text	My Label																										
background																											
text size	26																										
text colour																											
border	No																										
x	19.875																										
y	5.4																										
	<table><tr><th>Property</th><th>Value</th></tr><tr><td>type</td><td>shape</td></tr><tr><td>name</td><td>myShape1</td></tr><tr><td>x</td><td>21.175</td></tr><tr><td>y</td><td>11.475</td></tr><tr><td>speed</td><td>0</td></tr><tr><td>size</td><td>3</td></tr><tr><td>sides</td><td>3</td></tr><tr><td>colour</td><td></td></tr><tr><td>angle</td><td>0</td></tr></table>	Property	Value	type	shape	name	myShape1	x	21.175	y	11.475	speed	0	size	3	sides	3	colour		angle	0	 <p>The options offered will depend upon the property selected.</p>					
Property	Value																										
type	shape																										
name	myShape1																										
x	21.175																										
y	11.475																										
speed	0																										
size	3																										
sides	3																										
colour																											
angle	0																										
	<table><tr><th>Property</th><th>Value</th></tr><tr><td>type</td><td>vehicle</td></tr><tr><td>name</td><td>myVehicle1</td></tr><tr><td>x</td><td>5.175</td></tr><tr><td>y</td><td>15.625</td></tr><tr><td>allow off screen</td><td>No</td></tr><tr><td>rotation style</td><td>Face the angle</td></tr><tr><td>angle</td><td>0</td></tr><tr><td>speed</td><td>0</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>image</td><td></td></tr><tr><td>show/hide</td><td>show</td></tr></table>	Property	Value	type	vehicle	name	myVehicle1	x	5.175	y	15.625	allow off screen	No	rotation style	Face the angle	angle	0	speed	0	scale	100	image		show/hide	show	 <p>The options offered will depend upon the property selected.</p>	
Property	Value																										
type	vehicle																										
name	myVehicle1																										
x	5.175																										
y	15.625																										
allow off screen	No																										
rotation style	Face the angle																										
angle	0																										
speed	0																										
scale	100																										
image																											
show/hide	show																										


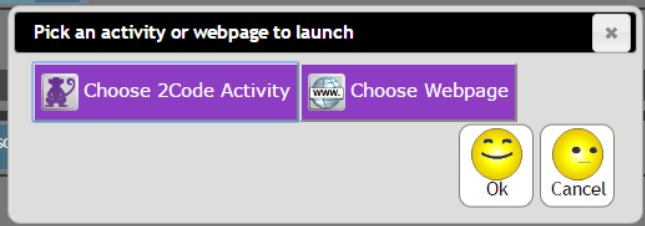
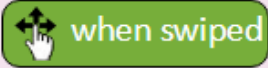
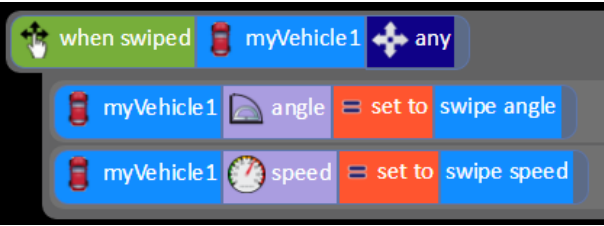
Object	Properties in Design View	Properties in Code View	Actions in code view																						
<div><p>character</p></div> <div><p>food</p></div> <div><p>animal</p></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>character</td></tr><tr><td>name</td><td>myCharacter1</td></tr><tr><td>x</td><td>32.925</td></tr><tr><td>y</td><td>4.175</td></tr><tr><td>movement</td><td>Stopped</td></tr><tr><td>allow off screen</td><td>No</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>speed</td><td>2</td></tr><tr><td>image</td><td></td></tr><tr><td>show/hide</td><td>show</td></tr></tbody></table>	Property	Value	type	character	name	myCharacter1	x	32.925	y	4.175	movement	Stopped	allow off screen	No	scale	100	speed	2	image		show/hide	show	<div></div>	<div></div> <p>These make the object move in different directions.</p> <p>Stop, hide or show the object.</p> <p>Make the object speak by displaying a speech bubble.</p>
Property	Value																								
type	character																								
name	myCharacter1																								
x	32.925																								
y	4.175																								
movement	Stopped																								
allow off screen	No																								
scale	100																								
speed	2																								
image																									
show/hide	show																								
<div><p>turtle</p></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>turtle</td></tr><tr><td>name</td><td>myTurtle1</td></tr><tr><td>x</td><td>10.825</td></tr><tr><td>y</td><td>15.95</td></tr><tr><td>angle</td><td>0</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>image</td><td></td></tr><tr><td>show/hide</td><td>show</td></tr></tbody></table>	Property	Value	type	turtle	name	myTurtle1	x	10.825	y	15.95	angle	0	scale	100	image		show/hide	show	<div></div>	<div></div> <p>A turtle moves in a similar way to a floor turtle using Logo type actions.</p> <p>Turn is by a number of degrees.</p>				
Property	Value																								
type	turtle																								
name	myTurtle1																								
x	10.825																								
y	15.95																								
angle	0																								
scale	100																								
image																									
show/hide	show																								

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
	<p>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:</p> 
	<p>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.</p>
	<p>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.</p>
	<p>This runs the code inside the first block if a certain condition is met, otherwise it runs the code inside the second block.</p> 
	<p>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.</p>
	<p>This will restart the program from the beginning. Useful if you want to include a restart button in your program.</p>

Command	Explanation
	<p>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.</p> 
	<p>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.</p> 

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

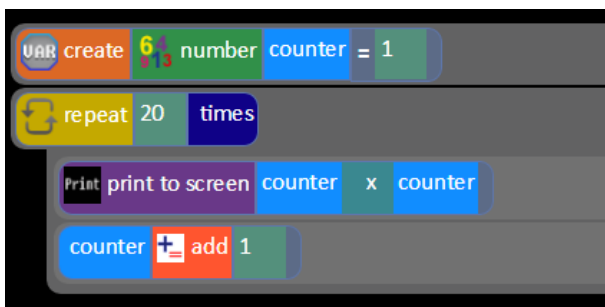


 set to or . If the variable is a text variable, the add command will concatenate the text. If it is a number it will add the new number on.

The following code will produce a list of random sentences:



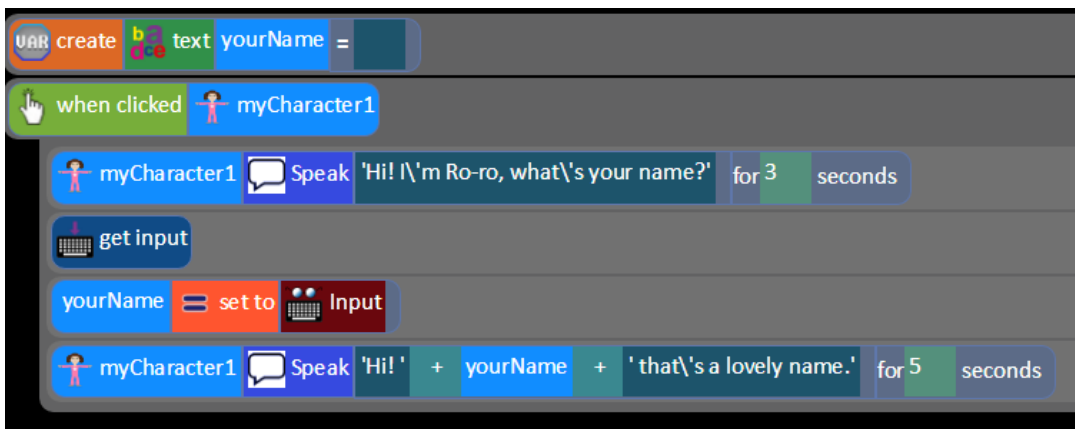
This code will produce the first 20 square numbers:



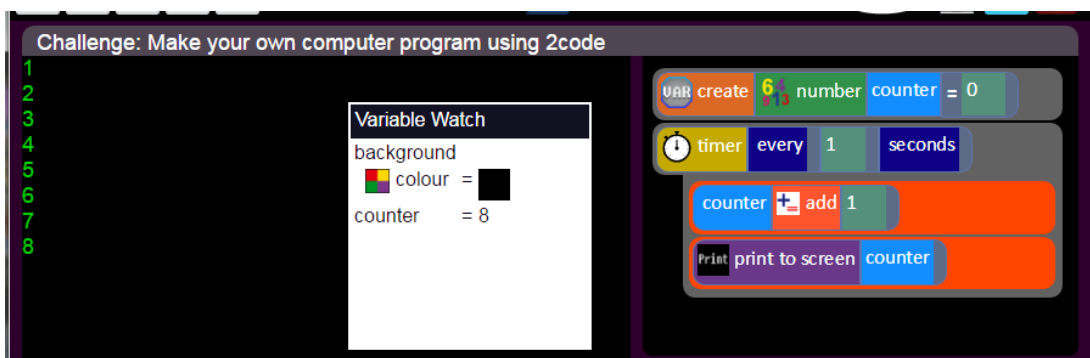
Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)

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.

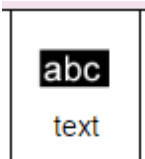
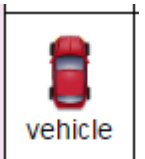





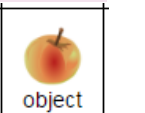








For example, in the Gorilla level, the Animal object doesn't have direction because it has angle, which is a different form of movement to the Character Object and why we don't have one button for both types of object.

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.

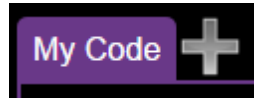
This creates an interesting variety of movement styles. Try animating your Character and your Animal up a slope – the concept of “up”, “down”, “left” and “right” (Character) don't work for this but rotating

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																										
<div></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>text</td></tr><tr><td>name</td><td>myText1</td></tr><tr><td>background</td><td></td></tr><tr><td>border width</td><td>0</td></tr><tr><td>border colour</td><td></td></tr><tr><td>text size</td><td>26</td></tr><tr><td>text colour</td><td></td></tr><tr><td>text align</td><td>left</td></tr><tr><td>font</td><td>sans-Serif</td></tr><tr><td>x</td><td>8.822</td></tr><tr><td>y</td><td>6.381</td></tr><tr><td>show/hide</td><td></td></tr></tbody></table>	Property	Value	type	text	name	myText1	background		border width	0	border colour		text size	26	text colour		text align	left	font	sans-Serif	x	8.822	y	6.381	show/hide		<div><div>texttext colourfontx y</div><div>show/hide</div></div>	Show/Hide
Property	Value																												
type	text																												
name	myText1																												
background																													
border width	0																												
border colour																													
text size	26																												
text colour																													
text align	left																												
font	sans-Serif																												
x	8.822																												
y	6.381																												
show/hide																													
<div></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>vehicle</td></tr><tr><td>name</td><td>myVehicle1</td></tr><tr><td>x</td><td>8.921</td></tr><tr><td>y</td><td>13.336</td></tr><tr><td>allow off screen</td><td>No</td></tr><tr><td>rotation style</td><td>Face the angle</td></tr><tr><td>angle</td><td>0</td></tr><tr><td>speed</td><td>0</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>image</td><td></td></tr><tr><td>friction</td><td>0</td></tr><tr><td>show/hide</td><td>show</td></tr></tbody></table>	Property	Value	type	vehicle	name	myVehicle1	x	8.921	y	13.336	allow off screen	No	rotation style	Face the angle	angle	0	speed	0	scale	100	image		friction	0	show/hide	show	<div><div>x yangle speed scale</div><div>imagefrictionshow/hide</div></div>	<div></div>
Property	Value																												
type	vehicle																												
name	myVehicle1																												
x	8.921																												
y	13.336																												
allow off screen	No																												
rotation style	Face the angle																												
angle	0																												
speed	0																												
scale	100																												
image																													
friction	0																												
show/hide	show																												
<div><div></div><div></div></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>character</td></tr><tr><td>name</td><td>myCharacter1</td></tr><tr><td>x</td><td>22.362</td></tr><tr><td>y</td><td>3.549</td></tr><tr><td>movement</td><td>Stopped</td></tr><tr><td>allow off screen</td><td>No</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>speed</td><td>2</td></tr><tr><td>friction</td><td>0</td></tr><tr><td>image</td><td></td></tr><tr><td>show/hide</td><td>show</td></tr></tbody></table>	Property	Value	type	character	name	myCharacter1	x	22.362	y	3.549	movement	Stopped	allow off screen	No	scale	100	speed	2	friction	0	image		show/hide	show	<div><div>x yscale speedfriction</div><div>image</div></div>	<div><div><div>updownleftright</div><div>stophide showSpeak</div></div></div>		
Property	Value																												
type	character																												
name	myCharacter1																												
x	22.362																												
y	3.549																												
movement	Stopped																												
allow off screen	No																												
scale	100																												
speed	2																												
friction	0																												
image																													
show/hide	show																												
<div></div>	<table><thead><tr><th>Property</th><th>Value</th></tr></thead><tbody><tr><td>type</td><td>character</td></tr><tr><td>name</td><td>myCharacter1</td></tr><tr><td>x</td><td>22.362</td></tr><tr><td>y</td><td>3.549</td></tr><tr><td>movement</td><td>Stopped</td></tr><tr><td>allow off screen</td><td>No</td></tr><tr><td>scale</td><td>100</td></tr><tr><td>speed</td><td>2</td></tr><tr><td>friction</td><td>0</td></tr><tr><td>image</td><td></td></tr><tr><td>show/hide</td><td>show</td></tr></tbody></table>	Property	Value	type	character	name	myCharacter1	x	22.362	y	3.549	movement	Stopped	allow off screen	No	scale	100	speed	2	friction	0	image		show/hide	show	<div><div>x yangle speed scale</div><div>imagefrictionshow/hide</div></div>	<div></div>		
Property	Value																												
type	character																												
name	myCharacter1																												
x	22.362																												
y	3.549																												
movement	Stopped																												
allow off screen	No																												
scale	100																												
speed	2																												
friction	0																												
image																													
show/hide	show																												

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 as "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.

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: [@2simplesoftware](#)

3.1 Scores

As with all Purple Mash activities, guided lessons can be set as 2Dos and allocated judgements aligned to the curriculum. Children's work can then be marked and assessed in the same way as with any piece of work. For details about making judgements for 2Dos see the [Data Dashboard User Guide](#).

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 Data Dashboard. For more information see the [Data Dashboard User Guide](#).

Pupil Name	Class Name	Score	Time
	Averages	4.00 Stars	56s
Mason Neptunium	Dolphin	2.00 Stars +	54s
Freya Carbon	Dolphin	3.00 Stars +	1m 5s
Joshua Bohrium	Dolphin	4.00 Stars +	57s
Jasper Cobalt	Dolphin	5.00 Stars +	40s
Layla Bismuth	Dolphin	5.00 Stars +	45s
Florence Polonium	Dolphin	5.00 Stars +	1m 16s



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.

My 2Code Scores		
2Code		
Activity	Stars	Time
a few seconds ago Fun with fish	★★★★★	00:44
12 days ago Fun with fish	★★★★★	05:29
17 days ago		

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](#).

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.

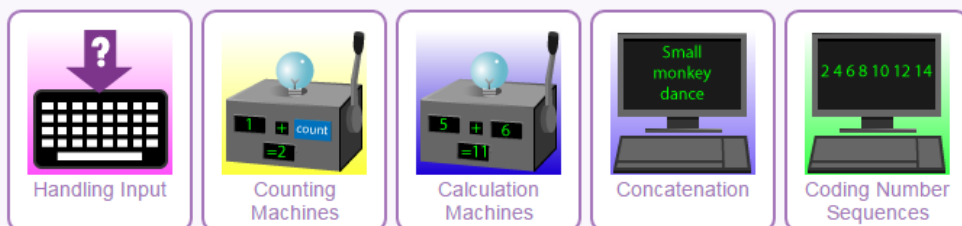
They explore coding functionality and coding structure that is common in many coding languages used to write programs that we use.

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:

Coding Principles:



Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: [@2simplesoftware](#)

7 Assessments

There are assessment activities for each level of 2Code; Chimp, Gibbon and Gorilla which are structured like the guided activities and assess children's skills.

The aim of the assessment activities is to find out how well children have understood the coding skills that they have been learning about and creating as part of the guided lessons. They can also be applied to the scheme of work as end of Y2, Y4 and Y6 assessments.

To enhance children's ability to code, understand the process of coding and to maximise their ability to complete the assessments successfully, children should have had as many of the following experiences as possible:

Challenges

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

Free coding

Children who have not had experience of 2Code free coding, for example those who have only used the guided lessons and not the scheme of work or other coding lessons will benefit from spending some time using:

- Y1-2 Free code Chimp (or Free code scenes)
- Y3-4 Free code Gibbon
- Y5-6 Free code Gorilla

To create their own programs.

Program Design

To master coding skills, children need to have the opportunity to explore program design and put computational thinking into practice. They could do this through:

- 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 time-line 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 – inputs and outputs)
- 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

7.1 Skills Assessed by Level

The following tables show the 2Code skills and coding concepts that are specifically covered by each of the guided lessons, these are the skills being assessed in the assessment task.

7.1.1 Skills Covered - Chimp

Activity	Coding Knowledge Introduced (all activities also revise previous knowledge)
Fun with Fish	Character objects Moving left and right
Bubbles	Click actions Moving up and down
Air Traffic Control	Vehicle objects*
Snail Race	Background actions Use numbers for speed Random numbers
Vehicles	Vehicle objects*

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Activity	Coding Knowledge Introduced (all activities also revise previous knowledge)
Turtle	Turtle object; moving and turning Collision detection Swipe action
Haunted Scene	Combining several objects and actions in a program Change image
Guard the Castle	Collision detection
Princess and the Frog	Timer for delay Stopping objects
Sounds	Sound function Play sounds using click and swipe.
Tick Tock Challenge	Timer for measuring time
Magician	Hide/ShowHide/Show
Jumping Monkey	Combining objects and actions in a narrative
Super Heroes	Scale property
Sparklers	Rotation
Rockets	Combining objects and actions and timers to make an animation
Night and Day	Use a timer to change the look of the screen (background)
Newton	Combining objects and actions in timers
Printing to the screen	Print to screen repeat

*Not assessed as these objects are not usually part of Chimp coding

7.1.2 Skills Covered - Gibbon

Activity	Coding Knowledge Introduced (all activities also revise previous knowledge)
Shapes	Shape object - sides property change to random colour when clicked size property + 1 sides property + 1
Random Words and Wizards	generating random words using the timer
Traffic Lights	decomposition and abstraction
Vehicles 2	Vehicle object angle speed If statements whenKey
Guard the Castle 2	Combining a timer and an if statement to check character position every second and do something.
Genie	Number variables
Switching backgrounds	Applying and practising variables
Night and Day	Applying and practising variables
Catherine Wheel	Rotation
Functions	Functions
Splatty Bug	Alert Scorepad x and y coordinates
Metric Conversions	Buttons

7.1.3 Skills Covered - Gorilla

Activity	Coding Knowledge Introduced (all activities also revise previous knowledge)
Send the Rocket to Space	Make a countdown timer with if/else inside it to stop countdown
Catching Game	Setting initial values (speed) in design mode
2Go	Use buttons Create and use functions Use input box data e.g. number of steps to go
Turtle Crossing Road	keypress for directions collision detection
Feed the Duck	Object speed keyboard and coordinate commands to control the objects scale in response to collision angle with keypress
Helicopter Swipe Game	Swipe speed and swipe angle properties Set random directions using the timer to change direction Collision detection Set x and y coordinates of objects to random
Dancer	Make disco effect by changing background colour every half second Use buttons to change object properties
Driving game	Walls racing game comparing numbers

Activity	Coding Knowledge Introduced (all activities also revise previous knowledge)
Football game	Control using swipe. Code friction into movement. Code collision detection. Add code to show the number of goals scored. Create a function to reset the game. Use collision detection to multiply speed of object by -1 to change direction decomposition and abstraction

7.2 Conducting the Assessments

The Assessments should be accessed from the Teacher Area --> User Guides and Planning --> Coding Resources --> [Guided Lessons and Solutions](#)

Set the required assessment as a 2Do for your pupils. Pupils will then be able to access the assessment. An assessment should be completed in a single session and children should save their file at the end of the assessment so that you can view their challenge activities.

7.3 Tasks and Solutions

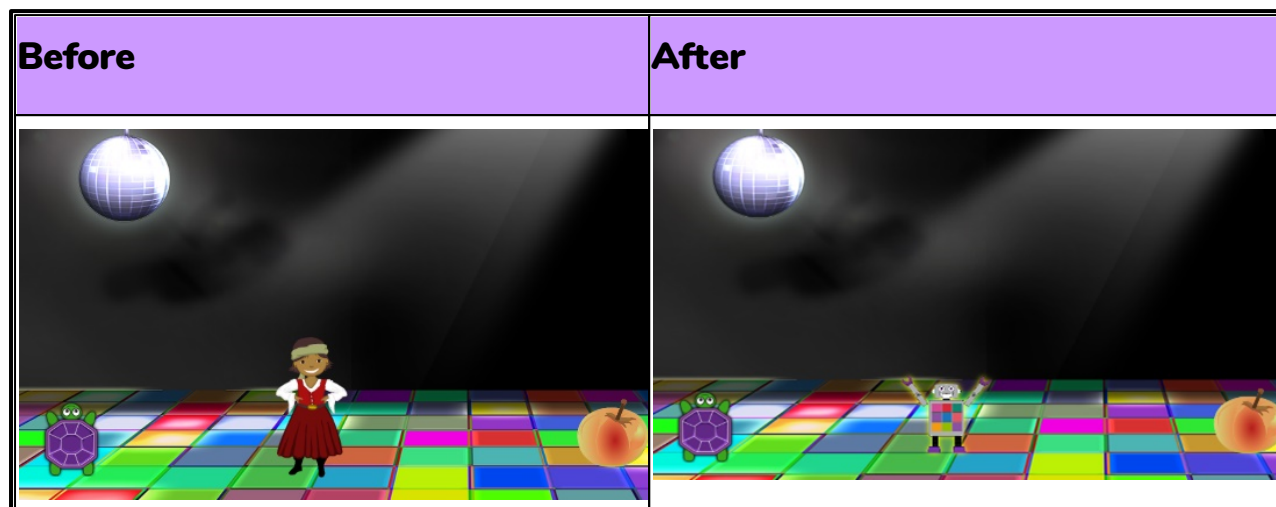
7.3.1 Chimp 1 - Purple Disco

Task 1



Instructions	Hint given	Skills Assessed
<p>The first task is to change the background to the disco background.</p> <p>Press play when you are done.</p>	<p>Reminder of where the background button is and to look for the image property.</p>	<p>Change backgrounds</p>

Task 2



Instructions	Hint given	Skills Assessed
Change the character's image to this robot (any dance position is fine). You will find it in the robot section in the clip-art picker.	Reminder to double-click on the character then navigate the clip-art picker.	Character image change using the clip-art picker.

Task 3



Instructions	Hint given	Skills Assessed
Change the turtle image to a different robot and the apple image to this pile of bricks found in the toys section of the clipart library.	Reminder to double-click on the character.	Object image change using the clipart picker

Task 4

Code



Instructions	Hint given	Skills Assessed
<p>Now make the robot in the middle move left and right changing direction when he collides with the bricks or the other robot.</p> <p>You will need to exit from design view for this.</p>	<p>Code view</p> <p>Collision detection</p>	<p>Moving</p> <p>Collision detection</p>

Task 5

Code



Instructions	Hint given	Skills Assessed
Now make dance robot change image to one of the other dance images for the robot when he collides with the objects.	Editing the code in the right place.	Editing code to add image changes when required.

Challenge \ Mastery open-ended task

All Chimp objects and code blocks are available to use for this.

Instructions

Make your disco more exciting.

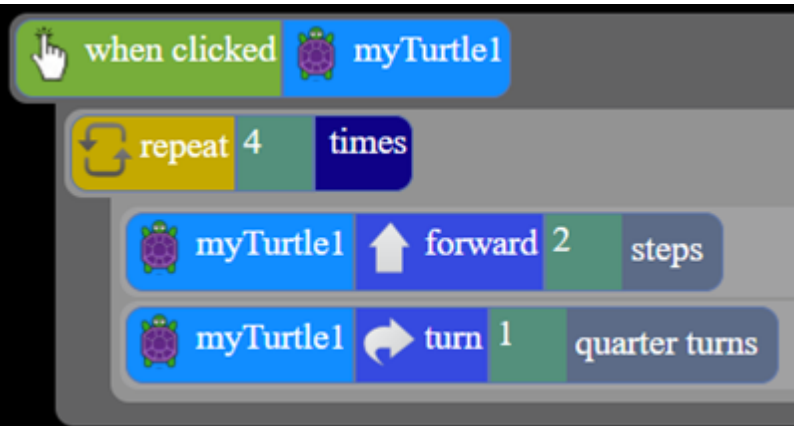
How about adding some more dancers and some music to your disco?

Children's opportunity to show their deeper understanding and creativity with the tools.

7.3.2 Chimp 2 - Turtle's Day Out

Task 1

Code



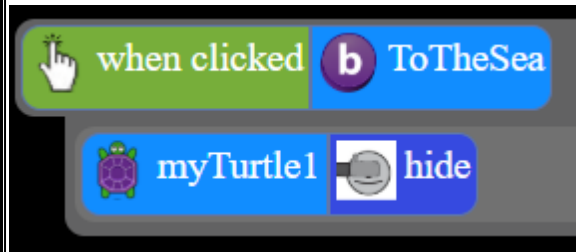
The code blocks are as follows:

- when clicked** (green flag icon)
- myTurtle1** (blue block with turtle icon)
- repeat** 4 **times** (yellow block with loop icon)
- myTurtle1** **forward** 2 **steps** (blue block with turtle icon and up arrow)
- myTurtle1** **turn** 1 **quarter turns** (blue block with turtle icon and right arrow)

Instructions	Hint given	Skills Assessed
<p>Turtle has swum to the seaside for a day out.</p> <p>First, he wants to draw some patterns in the sand.</p> <p>Write some code so that when he is clicked on, he draws a square with sides length 2 steps. You MUST use the repeat command in your code.</p> <p>When you test your code, use the pen down button to make him put his pen down before you click on him.</p>	<p>How to structure the repeat.</p>	<p>whenClicked command</p> <p>Repeat command</p> <p>Turtle properties</p>

Task 2

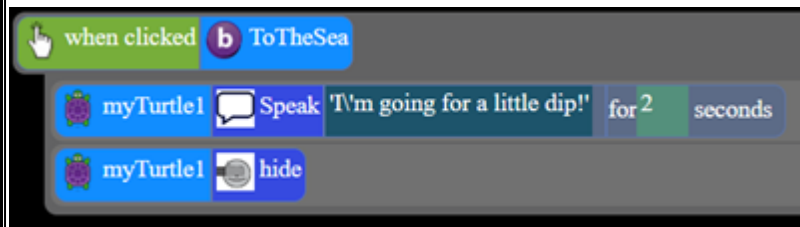
Code



Instructions	Hint given	Skills Assessed
Add some code so that when the button 'To the sea' is clicked, the turtle hides.	When clicked reminder	Hide property When clicked command

Task 3

Code



Instructions	Hint given	Skills Assessed
Add some code so he says, 'I'm going for a little dip', for 2 seconds, then he hides. (Note for teacher: actual text entered isn't tested.)	Timing of the command	Say property

Task 4

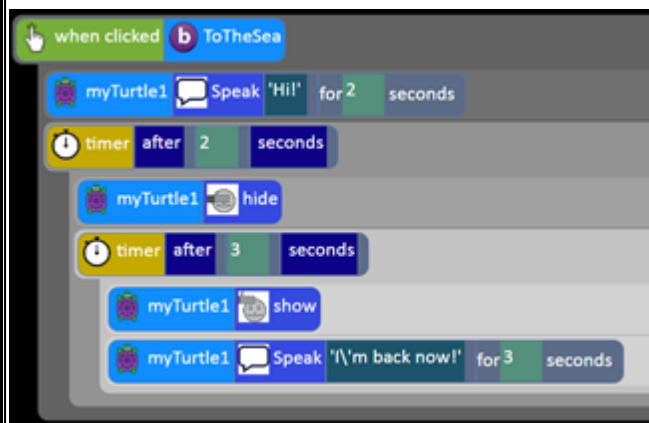
Code



Instructions	Hint given	Skills Assessed
Add a timer so he doesn't hide straight away, he hides after 2 seconds.	Structure of the timer command (after)	Timer - after

Task 5

Code



Instructions	Hint given	Skills Assessed
Add another timer inside the first one so he reappears after 3 seconds and says 'I'm back now!' (Note for teacher: actual text entered isn't tested. Children might need to be shown how to add an apostrophe in 'I'm'. The	Nesting a timer	Nesting a timer

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Instructions	Hint given	Skills Assessed
total time is 5 seconds hence the second timer being after 3 seconds (3+2=5)).		

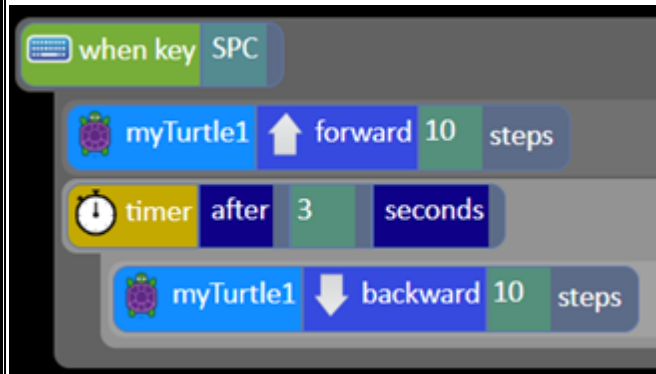
Task 6

Code


Instructions	Hint given	Skills Assessed
<p>Make turtle go to the gift shop and then back to his starting place when the player presses the spacebar.</p> <p>Turn the grid on by clicking the button so that you can easily count the steps.</p> <p>He should go on top of the other buildings in a straight line.</p>	<p>Design view and grid button</p> <p>(Note for teacher: after dragging in the when key command pupils should press the spacebar to insert the 'SPC' into the command.)</p>	<p>Design view</p> <p>When key pressed event</p>

Task 7

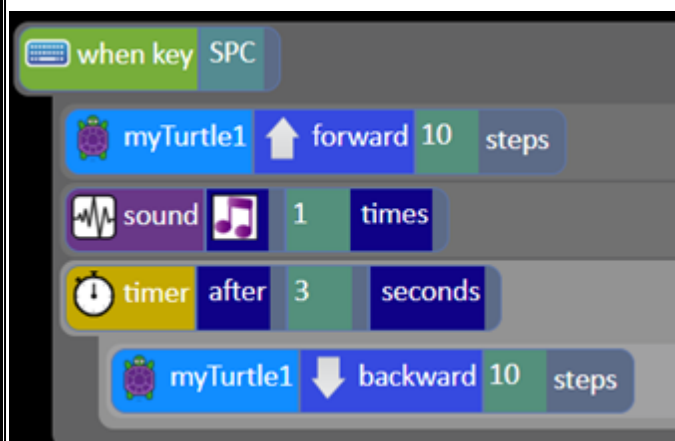
Code



Instructions	Hint given	Skills Assessed
He is in too much of a rush, add a timer so that he only comes back after 3 seconds.	Structure required – where to out the code	Editing code to add a timer

Task 8

Code



Instructions	Hint given	Skills Assessed
Make a chime sound play when he arrives at the gift shop	Sound command	Adding sound

Challenge \ Mastery open-ended task

All Chimp objects and code blocks are available to use for this.

Instructions

Add buttons to make turtle visit other places.

Make turtle draw something different in the sand.

Add a second turtle and make it perform some actions such as moving and talking.

Add different sounds

Children's opportunity to show their deeper understanding and creativity with the tools.

7.3.3 Gibbon 1 - Shape Game

Task 1

Before	After
	

Instructions	Hint given	Skills Assessed
<p>The current screen has a number object for the player score and a label for this box. Add a number box for the computer score called scorepadComp and a label 'Computer' for the score.</p> <p>You can look at the design view to do this."</p>	<p>Switching to design view.</p> <p>Adding objects in design view and renaming.</p>	<p>Adding and naming objects</p>

Task 2

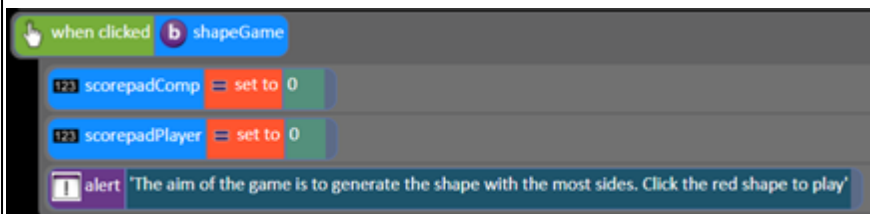
Code



Instructions	Hint given	Skills Assessed
<p>Drag in two shape objects for the shape game.</p> <p>Name them playerShape and compShape.</p> <p>Make playerShape red and compShape purple.</p> <p>(Note for teacher: if you have children who have colour-blindness in your class, they may need help to identify the colours).</p>	<p>Adding and naming an object.</p> <p>Colour property.</p>	<p>Adding and naming objects</p> <p>Adding shape objects</p> <p>Setting up the user interface.</p>

Task 3

Code




Instructions	Hint given	Skills Assessed
<p>Add code so that when the shapeGame button is clicked</p> <p>- it sets the scorepadComp and</p>	<p>Finding scorepadComp and initializing it.</p>	<p>Click events</p> <p>Resetting variables (initializing)</p>

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Instructions	Hint given	Skills Assessed
<p>scorepadPlayer to zero</p> <p>- it opens and alert that says:</p> <p>'The aim of the game is to generate the shape with the most sides. Click the player shape to play.'</p> <p>(Note for teacher: actual text entered isn't tested.)</p>		Alert control

Task 4

Code


Instructions	Hint given	Skills Assessed
<p>Add code so that when the playerShape is clicked it sets the number of sides of playerShape and compShape to a random number between 1 and 10.</p>		

Task 5

Code



Instructions	Hint given	Skills Assessed
<p>Inside the click event, add an if statement that tests whether the number of sides of the playerShape is greater than the number of sides of the compShape and adds one to the playerScorepad if it does.</p> <p>Add a similar statement that tests if the computer wins.</p>	If statements	<p>Selection; if statements</p> <p>Comparing properties of objects</p>

Task 6

Code

```

if playerShape 5 sides greater compShape 6 sides Then
  scorepadPlayer add 1
  print to screen "You win!"
if compShape 5 sides greater playerShape 6 sides Then
  scorepadComp add 1
  print to screen "Computer wins :-("
  
```

Instructions	Hint given	Skills Assessed
<p>Add print to screen commands that print 'You win!' or 'Computer wins :-(' depending upon the result.</p> <p>(Note for teacher: actual text entered isn't tested.)</p>	Where to add the alerts.	Print to screen using the value of other objects/properties

Challenge\ Mastery open-ended task

All Gibbon objects and code blocks are available to use for this.

Instructions

Can you make your game more exciting?

You could add more shapes, so the winner is the most sides of three shapes.

You could limit the minimum number of sides to 3 for each shape to make them easier to click on.

You could add the size property into the game, so the bigger shape also gets a point.

Children's opportunity to show their deeper understanding and creativity with the tools.

7.3.4 Gibbon 2 - Space Race


Task 1

Before	After
	

Instructions	Hint given	Skills Assessed
<p>Set your design view up so it looks like this image. You will need the following object types:</p> <ul style="list-style-type: none"> • a rocket which is a vehicle object • planet earth which is a food object • an astronaut which is a vehicle object • a star which is a food object. <p>You will find the images in the clipart picker in the space category.</p> <p>You must call the objects 'star', 'earth', 'astronaut' and 'rocket'.</p> <p>Press play when you are done.</p> <p>(Note for teacher: pupils might also want to resize the earth object to cover up the moon image.)</p>	<p>Switch to design view, add object, change images</p>	<p>Adding and naming objects.</p>

Task 2

Code

 alert 'Launch the rocket back to Earth, will it get there? Press the space bar to start'


Instructions	Hint given	Skills Assessed
<p>Add an alert that opens at the beginning of the program and says 'Launch the rocket back to Earth, will it get there? Press the spacebar to start.'</p> <p>(Note for teacher: actual text entered isn't tested. Children need to click OK on the alert to mark the task and move to the next challenge.)</p>	Adding an alert.	Creating alerts

Task 3

Code

var create 6 number winLose = 0

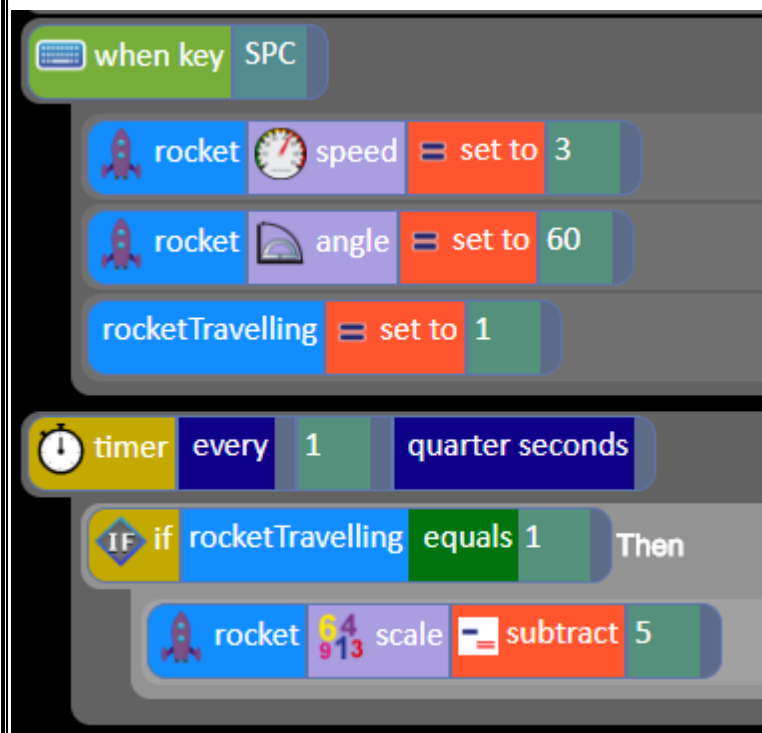
var create 6 number rocketTravelling = 0

 alert 'Launch the rocket back to Earth, will it get there? Press the space bar to start.'

Instructions	Hint given	Skills Assessed
<p>Create number variables called 'winLose' and 'rocketTravelling'. Set them to zero.</p> <p>(Note for teacher: Children need to click OK on the alert to mark the task and move to the next challenge.)</p>	How to create a variable and a its initial value.	Creating variables Initializing variables

Task 4

Code



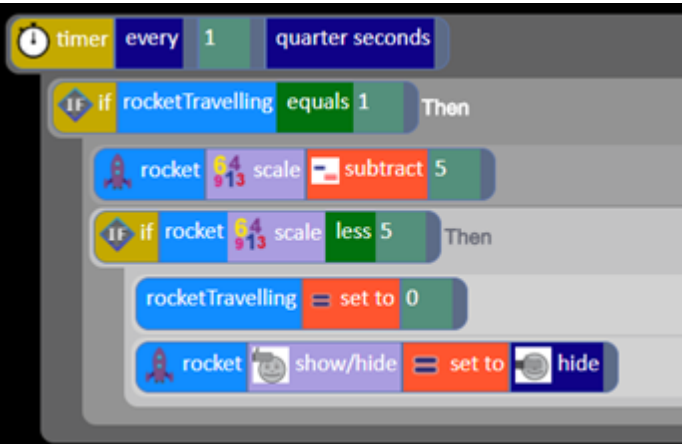
Instructions	Hint given	Skills Assessed
<p>The game will start when the spacebar is pressed. Add some code that runs when the spacebar is pressed that:</p> <ul style="list-style-type: none"> - Sets the rocket speed to 3 - sets the rocket angle so it will collide with Earth (you will need to test and debug this). - Use a timer to set the rocket scale to decrease by 5 every 1 quarter second so it looks like it is flying into the distance. <p>(Note for teacher: this is a tricky task because there are a few different things to do. Encourage the children to tackle one at a time).</p> <p>(Note for teacher: after dragging</p>	<p>Breaking the task down into parts.</p> <p>When key command.</p> <p>Timer</p> <p>Changing variable value</p>	<p>Key press event</p> <p>Setting object speed, scale and angle</p> <p>Using a timer</p>

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Instructions	Hint given	Skills Assessed
in the when key command pupils should press the spacebar to insert the 'SPC' into the command).		

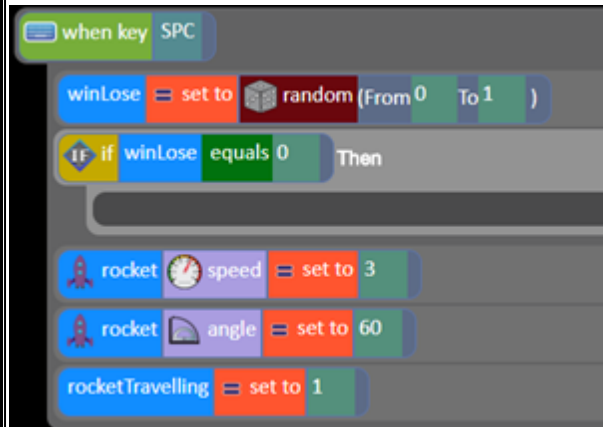
Task 5

Code


Instructions	Hint given	Skills Assessed
You might have noticed that the spaceship gets to 0 size and starts growing again. Let's stop that, shall we? Add an if statement after the scale is reduced. If the scale is less than 5, set the rocketTravelling variable to 0 and hide the rocket.	Editing the code – finding the right place. Changing variable values.	Changing variables Using Boolean comparison (less than) in an if statement.

Task 6

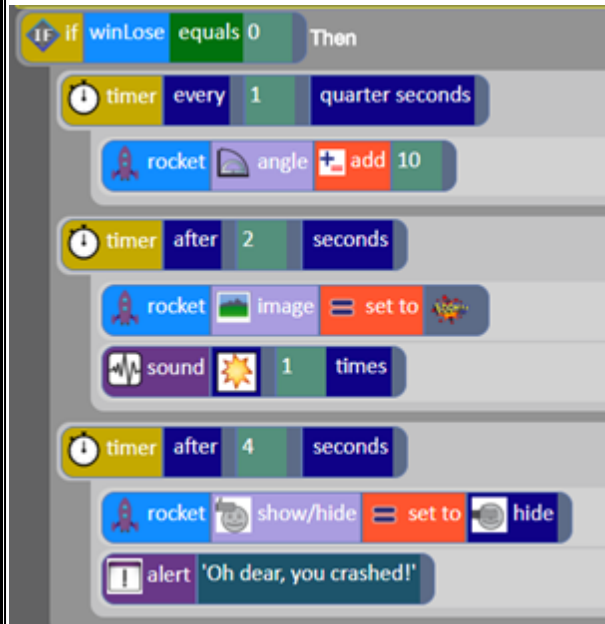
Code



Instructions	Hint given	Skills Assessed
<p>Whether the rocket makes it will be determined randomly. Add some code so that the variable winLose is set to a random number between 0 and 1 when the space bar is pressed.</p> <p>Add an if statement to test if winLose is 0.</p> <p>(Note for teacher: children do not need to put any code into the if statement on this step).</p>	<p>Editing the code:</p> <ul style="list-style-type: none"> Finding the right place. Setting variable to a random number. Adding an if statement. 	<p>Changing variables</p> <p>Using random function</p> <p>if statements</p> <p>Using binary values to determine outcomes (0=fail, 1=success).</p>

Task 7

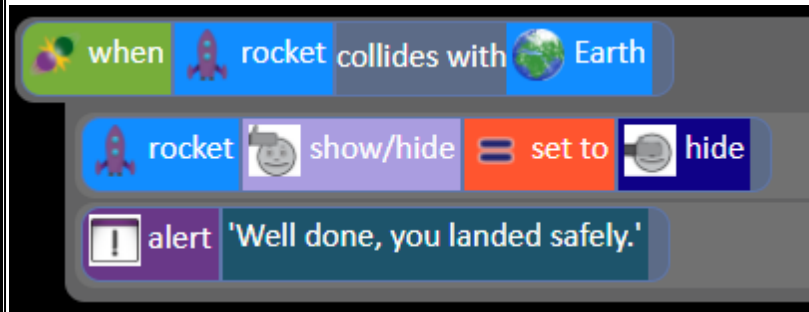
Code



Instructions	Hint given	Skills Assessed
<p>In the if statement add three timers that run if winLose=0:</p> <ul style="list-style-type: none"> • One that fires every 1 quarter second that adds 10 to the rocket's angle. • One that fires after 2 seconds that sets the rocket's image to an explosion (find this in the splats category of the clipart picker) and makes a bang sound. • One that fires after 4 seconds to hide the rocket and send an alert saying 'Oh dear! You crashed.' 	Where to add the code.	<p>Adding timers</p> <p>Timers and nested code.</p>

Task 8

Code



Instructions	Hint given	Skills Assessed
<p>Add collision detection that detects when the rocket collides with Earth, it should send an alert 'Well done, you landed safely.'</p> <p>Remove extra code that could stop the program working.</p> <p>Note to teacher: the steps to tidy up the code are shown in the task video.</p>	Where to add the alert.	<p>'Smelly' code</p> <p>Collision detection</p> <p>Alert</p>

Challenge\ Mastery open-ended task

All Gibbon objects and code blocks are available to use for this.

Instructions

What actions can you add to your space adventure?

Add some actions for the other objects; make the astronaut float around in space

Make the star pulse bigger and smaller.

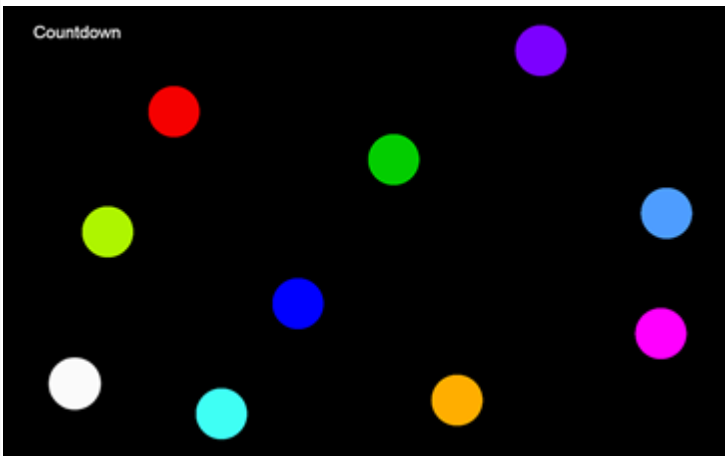
Add a reset button that sets the rocket back to the start to have another go.

Children's opportunity to show their deeper understanding and creativity with the tools

7.3.5 Gorilla 1 - Happy Eater Part 1

Task 1

Design



Instructions	Hint given	Skills Assessed
<p>Using vehicle objects, create 9 circles of different colours named 'catcher1' to catcher9'.</p> <p>Add a number object and call it 'Countdown'.</p> <p>Add a character object called 'player' and make it a green circle.</p> <p>(Note for teacher: The catchers must be vehicle objects with the images changed; circle images are found in the 'shapes' category in the clipart picker. The circles will be too large when first added so children can resize them by setting the scale in design view. Children can take a shortcut by creating one catcher called 'catcher1' with the appropriate image and scale then copy it 8 times; it will automatically rename the copies so only the image needs to be changed. Press Play to move to the next step.</p>	<p>Switch to design view.</p> <p>Adding objects, renaming and changing image and scale in design view.</p>	<p>Designing the GUI</p> <p>Naming objects appropriately.</p>

Task 2

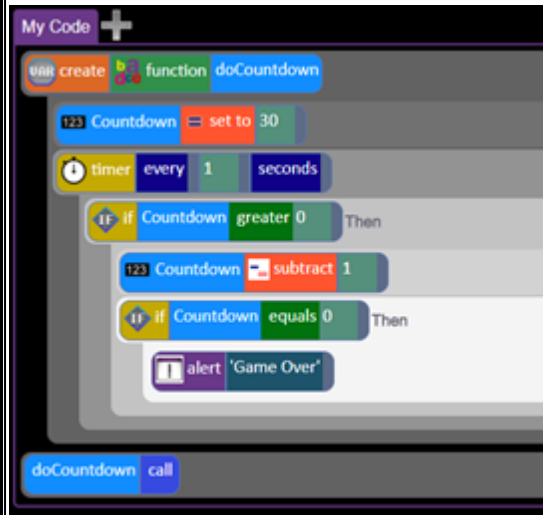
Code



Instructions	Hint given	Skills Assessed
<p>In code view, create a function called doCountdown. Inside the function create a timer that counts down from 30 and displays the result in the countdown box. The numbers should decrease by 1 every second.</p> <p>Call the function at the beginning of the game.</p>	<p>Creating a function</p> <p>Structuring a timer in code</p>	<p>Creating and calling functions.</p> <p>Making a countdown.</p> <p>Timer.</p> <p>Setting initial values.</p>

Task 3

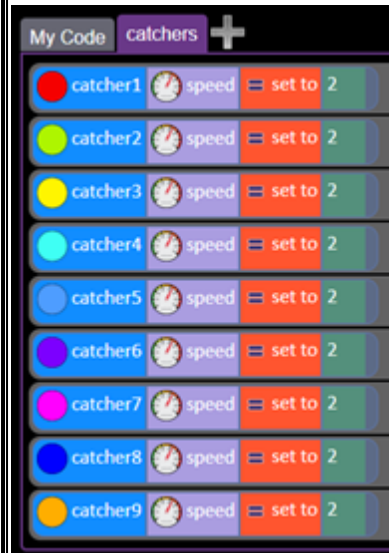
Code



Instructions	Hint given	Skills Assessed
Add code in the function timer that checks if countdown is greater than zero before subtracting 1. If countdown equals zero an alert 'Game over' should appear.	Structuring the code.	Use of an if statement with a timer to stop the countdown going below zero Nested if statement

Task 4

Code



Instructions	Hint given	Skills Assessed
In a separate tab called catchers, make all the catcher circles move at speed 2.	Adding a tab	Use of tabs Object speed property

Task 5

Code



Instructions	Hint given	Skills Assessed
In the Catchers tab, add a timer that fires every 5 seconds that sets the angle of each circle to a random number between -40 and 40 degrees.	Adding a timer, setting timer functions, structuring the code	Set random directions for multiple objects using the timer to change direction

Task 6

Code



Instructions	Hint given	Skills Assessed
<p>In the My Code tab, write code to make the player move in response to the direction keys being clicked or the screen being swiped so the game will work on a touchscreen as well as using a keyboard.</p> <p>Note to teacher: The task will pass if either the swipe code or the keypress code or both are input. If children are using tablets only then they will find it easier to just include the swipe code. A common error pupils make is coding for when the player is swiped rather than the background.</p>	When swiped and when key commands	Coding for different inputs including touch and key press.

Task 7

Code



Instructions	Hint given	Skills Assessed
<p>Add collision detection code for when player collides with any of the other circles (any vehicle).</p> <p>Inside the collision detection add an if/else statement that checks whether the scale of player is greater than the scale of the collided object.</p>	<p>Collision detection, selecting the 'Any Vehicle' object, structuring the if statement.</p>	<p>Coding for dependency using collision detection</p> <p>Comparing object properties</p>

Task 8

Code



Instructions	Hint given	Skills Assessed
<p>If player scale is bigger, then:</p>	<p>To code for the collided object,</p>	<p>Using if/else</p>

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Instructions	Hint given	Skills Assessed
<ul style="list-style-type: none"> • Add the scale of the collided object to the player scale to make the player increase in size. • Decrease the scale of the catcher (collided object) by 2. <p>If player scale is smaller, then:</p> <ul style="list-style-type: none"> • Decrease the player scale by 2. • Increase the scale of the catcher (collided object) by 2. <p>Note for teacher: the video is important to watch is it shows how to code for all vehicles when there doesn't appear to be the code block to do this.</p>	<p>drag any of the catchers (vehicles) into the code and then click on their name and you will see the option of 'collided vehicle'</p>	<p>scale in response to collision.</p> <p>coding for dependency using collision detection</p> <p>Comparing object properties</p>

Challenge\ Mastery open-ended task


All Gorilla objects and code blocks are available to use for this.

Instructions
<p>No specific task as this game is built on in the next assessment. To demonstrate mastery, pupils will be able to suggest some functionality that will be added in the next activity to finish and improve the game</p>

7.3.6 Gorilla 2 - Happy Eater Part 2

Task 1

Code



The code block shows a function named `resizeCatchers` that takes a parameter `catcher`. Inside the function, there are nine `scale = set to` blocks, each setting a different `catcher` (catcher1 through catcher9) to a specific scale value. Below the function, there are two `call` blocks: `resizeCatchers` and `doCountdown`.

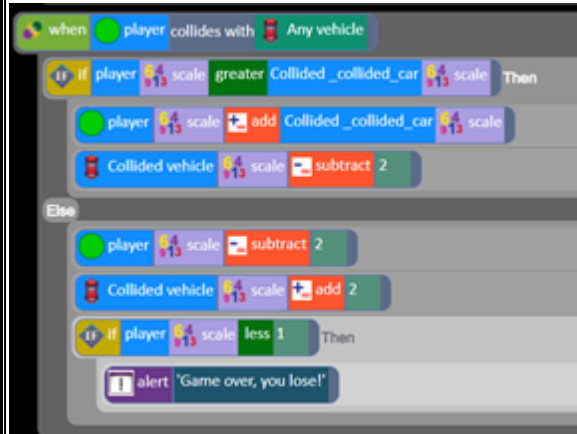
Instructions	Hint given	Skills Assessed
<p>Currently, all the circles start the same size. Add a function that resizes each of the shapes, make some smaller than the player and some larger.</p> <p>Call the function at the start of the game.</p> <p>Note to teacher: The specific sizes are not tested. Children should look at the scale of the player circle in design view to help them decide scales appropriately.</p>	<p>Creating a function</p> <p>Structure the function.</p>	<p>Call the function.</p> <p>Creating functions</p> <p>Calling functions</p>

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: [@2simplesoftware](https://twitter.com/2simplesoftware)

Task 2

Code



Instructions	Hint given	Skills Assessed
When the player decreases in scale add a check that ends the game if the player scale is less than 1.	Where to add the code. Structuring the required code.	Using if statements to perform an action dependent upon object properties.

Task 3

Code



Instructions	Hint given	Skills Assessed
When a catcher decreases in scale add a check that makes them hide	Locating the appropriate place to edit the code. Finding the	Using if statements to perform an action dependent upon object

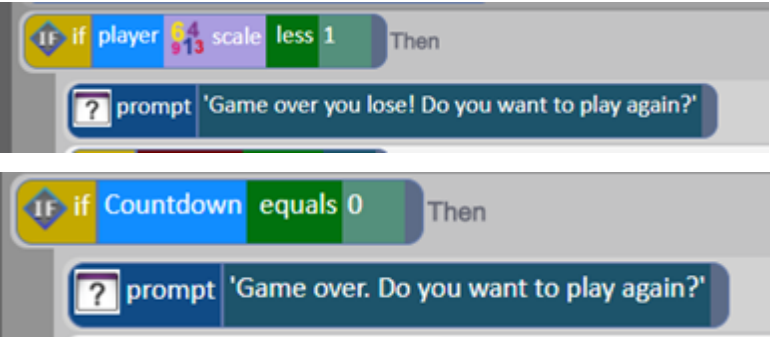
Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Instructions	Hint given	Skills Assessed
if their scale is less than 1.	'collided-vehicle' code block.	properties.

Task 4

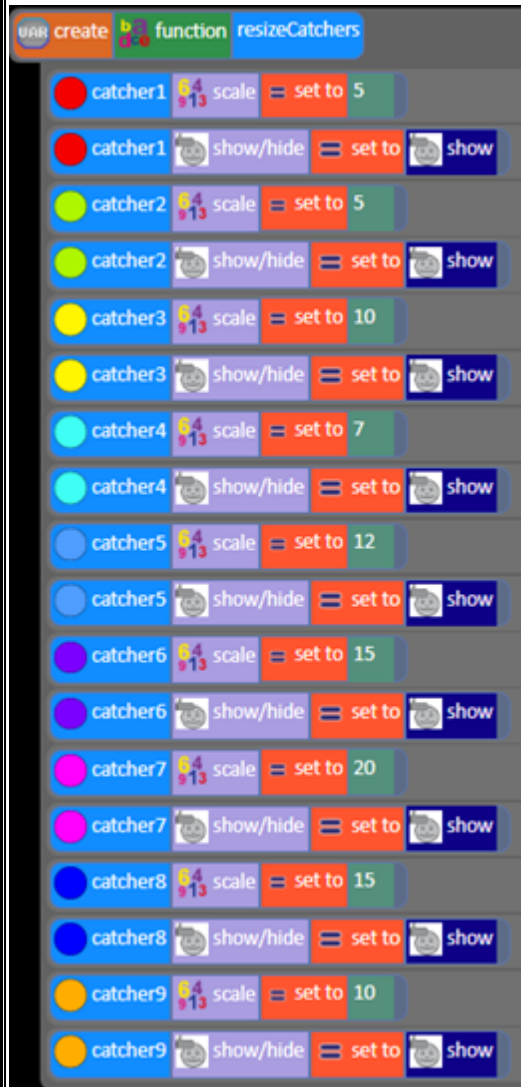
Code



Instructions	Hint given	Skills Assessed
<p>Add a way to restart the game by changing the alerts to prompts for input that ask the player if they want to start again?</p> <p>Note for teacher: The specific text of the alert isn't tested. Children might think that their code is not working because nothing happens when you answer the questions; this is an opportunity to remind the children that nothing happens because they haven't written the code yet, this will come in the next few steps.</p>	<p>Locating the code. Adding a prompt.</p>	<p>Resetting the game</p> <p>Taking user input</p>

Task 5

Code



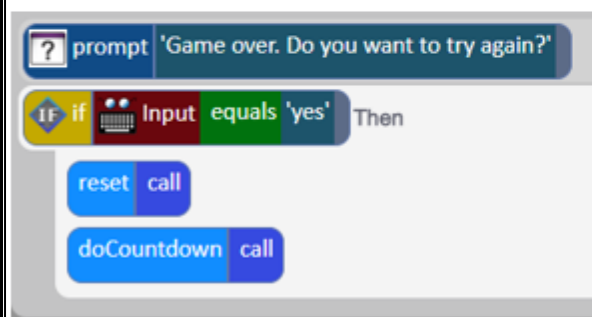
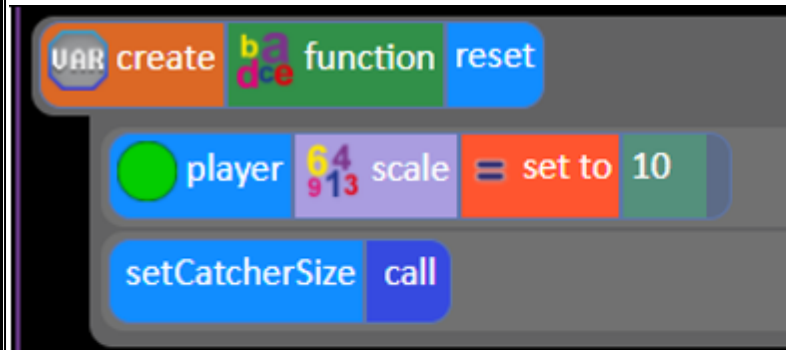
Instructions	Hint given	Skills Assessed
Edit the resizeCatchers function to make all the catchers visible.	Where to edit.	Resetting the game - reinitializing object properties.

Need more support? Contact us:

Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Task 6

Code



x 2 (once for
each prompt
box in the code)

Instructions	Hint given	Skills Assessed
<p>Create a new function called reset that calls resizeCatcher and also sets the player's scale back to 10.</p> <p>Call this function if the player enters 'yes' on the prompt boxes.</p> <p>Also call the function to reset the timer to 30.</p> <p>Note; a function can only call another function if the function appears above it in the code. You will need to make sure that your functions appear in the order: resizeCatcher, reset, doCountdown. You can drag them to the right place if they are in the wrong order.</p>	<p>Nested function</p> <p>When to call the function</p> <p>Which other function to call</p>	<p>Resetting the game - creating a reset function</p>

Need more support? Contact us:

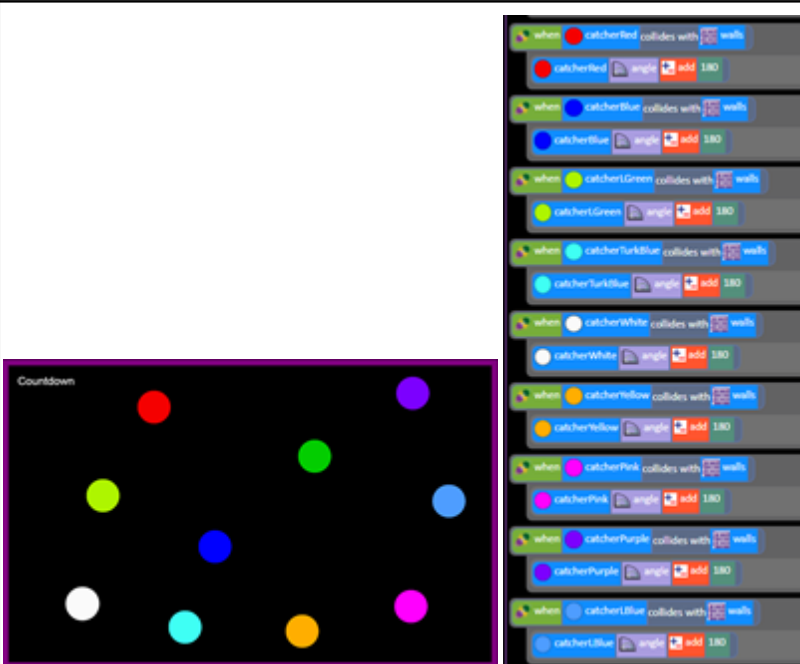
Tel: +44(0)208 203 1781 | Email: support@2simple.com | Twitter: @2simplesoftware

Instructions	Hint given	Skills Assessed
Note for teacher: remind children to use the undo button if they drag code and it disappears. Sometimes it gets dragged over the trash can by mistake.		

Challenge\ Mastery open-ended task

All Gorilla objects and code blocks are available to use for this.

Instructions
<p>Add some walls and write code so that the catcher circles 'bounce back rather than going off the screen.</p> <p>Try altering the speed of the player so that it can catch up with the catcher circles; does this make the game too easy?</p> <p>Add information about the player's scale at the end of the game - this could be in the alert message or a print to screen message and it should give the player an idea of how they have done, you could include information about the scale of the other remaining circles as well.</p>

Code
 <p>The screenshot shows a game scene on the left with a black background and a 'Countdown' label. It contains several colored circles (red, purple, green, blue, cyan, orange, pink, white) and a grey circle. On the right, a list of code blocks is shown, each starting with 'when [catcherRed/catcherBlue/catcherGreen/catcherTurkBlue/catcherWhite/catcherYellow/catcherPink/catcherPurple/catcherBlue] collides with [wall]'. The code blocks include 'angle' and 'add 180' blocks, indicating a 180-degree turn upon collision.</p>

8 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.

9 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.