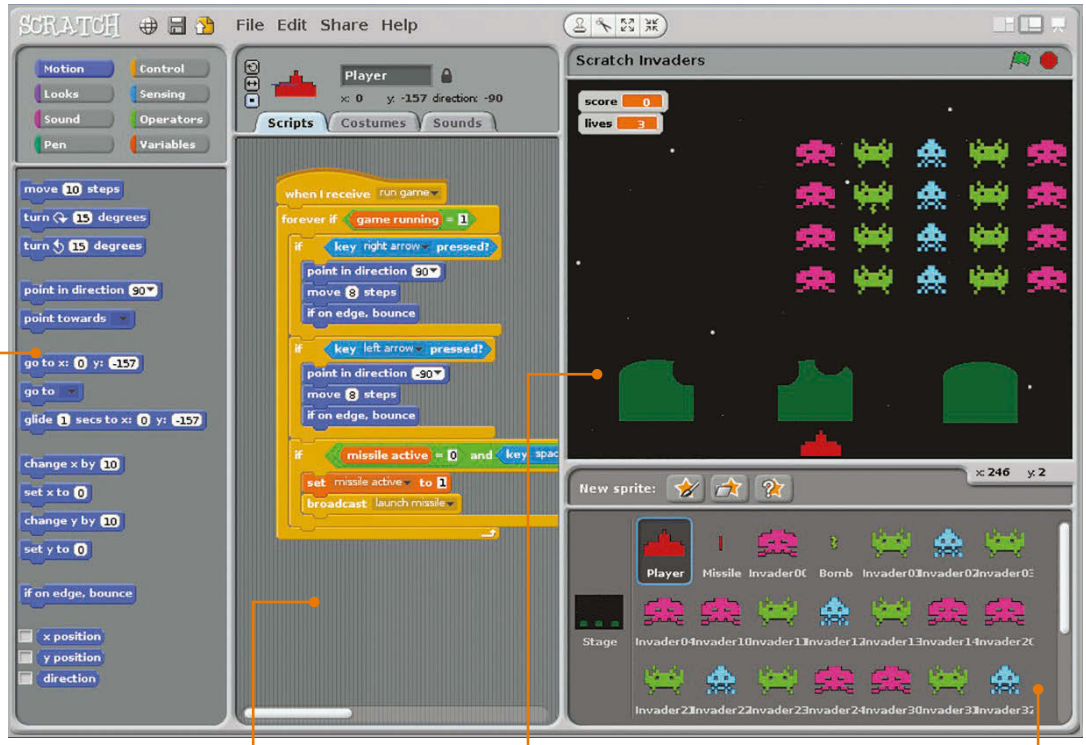
Blocks Palette: This contains blocks for programming, which you drag into the Scripts Area to add them to your code. There are eight colour-coded categories, selected from the top, each offering a different selection of blocks.



Scripts Area: This is the area where scripts are assembled. It can be accessed from sprites or the Stage, by selecting the Scripts tab. Note that you can create multiple scripts for each sprite. Click the tabs above to switch to costumes or sounds.

Stage: This is where your Scratch creations come to life. Sprites placed here can be resized using the grow and shrink icons above. Click the green flag to start the project running, and the red circle to stop it. There are also icons to change the view, including full-screen presentation mode.

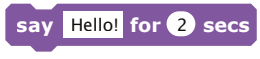
Sprite List: This contains thumbnails of all your sprites. Click one to select it and edit its scripts, costumes, and sounds. The icons above let you paint a new sprite, import one, or select a random one.

BLOCK SHAPES

Blocks are shaped according to the way in which they are used. There are six main types…

Hat Blocks: These are the Control blocks used to start every script – when the green flag is clicked, a key pressed, sprite clicked, or message received.

Stack Blocks: Shaped like jigsaw pieces to fit under and over others, these perform the main commands within scripts.



C Blocks: Generally resembling the letter C, these Control blocks can be wrapped around others to create loops or check for conditions.

Boolean Blocks: These hexagonal blocks contain conditions that, when invoked, report a value of true or false.

Reporter Blocks: Shaped with rounded edges, these hold values – numbers or strings. They include variables and lists.

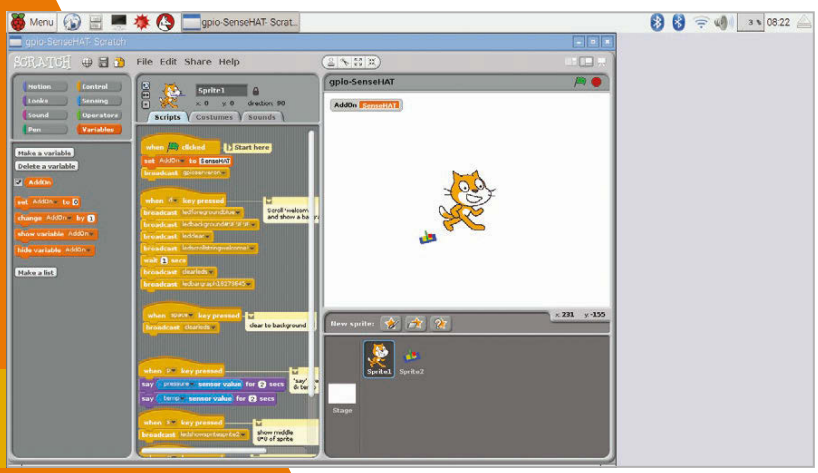
Cap Blocks: There are only two of these, found at the bottom of the Control category, used to stop one script or all of them.

SCRATCH GPIO

Scratch on the Pi now features a GPIO server for physical computing

In the latest version of Raspbian Jessie, Scratch features a Raspberry Pi GPIO server to make it easier to drive connected LEDs, buzzers, HATS, and other devices and components. First, you need to turn the server on via the Edit menu or running a **broadcast gpioserveron** block. You can then use **broadcast** blocks to configure and trigger individual GPIO pins, and use pulse-width modulation on pin 18. Other functions include taking a photo with the Camera Module, and obtaining the time and IP address. Certain Pi add-on boards and HATs are also supported, set up by creating an **AddOn** variable and setting it to the respective board name. For full details on this and other GPIO functionality, visit **magpi.cc/1TYX7Jg**.

Below: Using the Pi's GPIO pins



BLOCK REFERENCE GUIDE

A guide to all the blocks in each of the eight colour-coded categories, including tips for their usage…

Motion

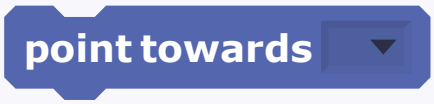
Motion blocks deal with the movement of sprites. They relate mainly to the x and y position and direction of the sprite.

Moves sprite forward by specified number of steps, or backwards (using a minus number). Useful for any project involving movement.

Rotates sprite clockwise by specified number of degrees.

Rotates sprite anticlockwise by specified number of degrees.

Points sprite in the specified direction: 0 = up, 90 = right, 180 = down, -90 = left. Other numbers may also be used.



Points sprite towards mouse pointer or another sprite. Can be used for steering a sprite with the mouse pointer.

Moves sprite to specified x- and y-position on stage. Useful for resetting its position at the beginning of a project.

Moves sprite to the location of the mouse pointer or another sprite. Useful for keeping a set of sprites together.

Moves sprite smoothly to a specified position over specified length of time. One downside is that it pauses the script while the sprite is gliding.

Changes sprite's x-position by specified amount. Often used in game controls.

Sets sprite's x-position to specified value. Can be used for horizontal scrolling.

Changes sprite's y-position by specified amount. Often used in game controls.

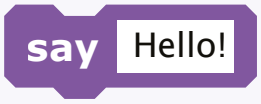
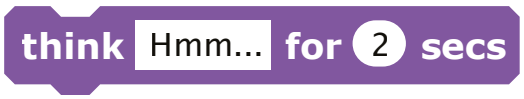
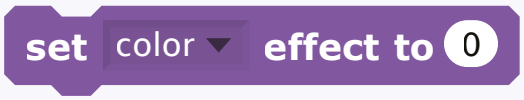
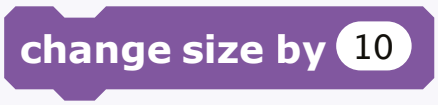
Sets sprite's y-position to specified value. Can be used for vertical scrolling.

if on edge, bounce	Turns sprite in opposite direction when it touches edge of stage. Handy for preventing it partially leaving the screen.
x position	Reports sprite's x-position (ranges from -240 to 240). Tick box to show on stage.
y position	Reports sprite's y-position (ranges from -180 to 180). Tick box to show on stage.
direction	Reports sprite's direction: 0 = up, 90 = right, 180 = down, -90 = left. Tick box to show on stage.

Looks

Looks blocks are used to control the appearance of sprites and the stage. Functionalities include changing costumes and applying graphic effects.

switch to costume costume1	Changes sprite's appearance by switching to different costume. Useful for animation.
next costume	Changes sprite's costume to next costume in the list (if at the end, it jumps back to first costume).
costume#	Reports sprite's current costume number. Tick box to show on stage.
say Hello! for 2 secs	Displays sprite's speech bubble for specified amount of time.

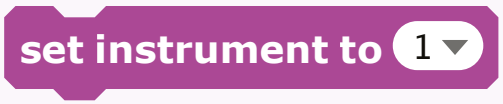
	Displays sprite's speech bubble. (To remove bubble, run this block without any text.)
	Displays sprite's thought bubble for specified amount of time.
	Displays sprite's thought bubble. (To remove bubble, run this block without any text.)
	Changes selected visual effect on a sprite by specified amount. Choose from colour, fisheye, whirl, pixelate, mosaic, brightness, and ghost effects.
	Sets selected visual effect to a given number.
	Clears all graphic effects for a sprite.
	Changes sprite's size by specified amount.
	Sets sprite's size to specified % of original size.
	Reports sprite's size as % of original size. Tick box to show on stage.
	Makes sprite appear on the stage (after being hidden).

hide	Makes sprite disappear from the stage. (Note that when a sprite is hidden, other sprites cannot detect it with a **touching?** block.)
go to front	Moves sprite in front of all other sprites. If it's large enough, it could cover the entire stage.
go back 1 layers	Moves sprite back a specified number of layers, so that it can be hidden behind other sprites.

Sound

These blocks are related to playing various sounds, which can be recorded or imported. 128 built-in MIDI instruments are also available.

play sound meow	Starts playing selected sound, selected from pull-down menu, and immediately goes on to the next block even as sound is still playing.
play sound meow until done	Plays a sound and waits until it has finished playing before continuing with next block.
stop all sounds	Stops playing all sounds.
play drum 48 for 0.2 beats	Plays selected drum sound for specified number of beats.

	Rests (plays nothing) for specified number of beats.
	Plays selected musical note for specified number of beats. (Clicking the pull-down arrow brings up a two-octave keyboard, but you can enter lower/higher numbers directly.)
	Sets the type of instrument that the sprite uses for **`play note`** blocks. (Each sprite has its own instrument.)
	Changes sprite's sound volume by specified amount. Volume ranges from 0 to 100.
	Reports sprite's sound volume. Tick box to show on stage.
	Changes sprite's tempo by specified amount (in beats per minute).
	Sets sprite's tempo to specified value in beats per minute.
	Reports sprite's tempo in beats per minute. Tick box to show on stage.

Pen

Pen blocks enable a sprite to draw lines and shapes, including its own 'stamp' image, on the stage when moved.

clear	Clears all pen marks and stamps from the stage.
pen down	Puts down sprite's pen, so it will draw as it moves.
pen up	Pulls up sprite's pen, so it won't draw as it moves.
set pen color to	Sets pen's colour, selected from colour picker. Picking the colour also changes the pen shade.
change pen color by 10	Changes pen's colour by specified amount.
set pen color to 0	Sets pen's colour to specified value (ranging from 0 to 200).
change pen shade by 10	Changes pen's shade (ranging from dark to light) by specified amount.
set pen shade to 50	Sets pen's shade to specified amount. It ranges from 0 (very dark) to 100 (very light). The default is 50, unless set with colour picker.
change pen size by 1	Changes thickness of pen line.

Sets thickness of pen line.

Stamps sprite's image onto the stage.

Control

Control blocks provide functions for looping scripts and only running them if certain conditions are met. The **broadcast** block can be used with the Raspberry Pi's GPIO pins.

Runs the script below once the green flag is clicked to start the project.

Runs script below when specified key is pressed. Useful for player controls in games.

Runs script below when sprite is clicked. Useful for menu buttons/options.

Waits specified number of seconds, then continues with next block. Use it whenever a pause is needed. It's not as accurate as using **timer**.

One of the most commonly used blocks, it runs the blocks inside it over and over, in a never-ending loop.

Runs the blocks inside a specified number of times. Common uses include sprite animation and movement.

Sends a message to all sprites, then continues with the next block without waiting for the triggered scripts. It can also be used to configure and trigger the Raspberry Pi's GPIO pins, and take a photo with the Pi Camera Module.

Sends a message to all sprites, triggering them to do something, and waits until they all finish before continuing with next block.

Runs the script below it when it receives specified broadcast message.

The equivalent of an **`if`** block within a **`forever`** one. Continually checks whether condition is true; whenever it is, it runs the blocks inside.

One of the most widely used blocks. If its condition is true, it runs the blocks inside.

If condition is true, runs the blocks inside the **`if`** portion; if not, runs the blocks inside the **`else`** portion.

Waits until condition is true, then runs the blocks below. Uses include waiting for a sprite to move somewhere, a value to pass a certain amount, or a reply from another script.

Checks to see if condition is false; if so, runs blocks inside and checks condition again. If condition is true, goes on to the blocks that follow.

Stops the script. Handy for disabling scripts, which can be restarted with a broadcast or key press.

Stops all scripts in all sprites. Can be used to end or pause a project.

Sensing

Sensing blocks can be used to detect when one sprite touches another. The **sensor value** block can be used to obtain a Pi GPIO pin's input.

Reports true if sprite is touching specified sprite, edge, or mouse pointer. Useful for collision detection in games.

Reports true if sprite is touching specified colour (selected using eyedropper). Again, handy for collision detection.

Reports true if first colour (within sprite) is touching second colour (in background or another sprite). Both colours are selected using eyedropper.

Asks a question on the screen and stores keyboard input in **answer**. Causes the program to wait until the **ENTER** key is pressed or checkmark is clicked.

Reports keyboard input from most recent use of **ask and wait** (shared by all sprites).

Reports the x-position of mouse pointer.

Reports the y-position of mouse pointer.

	Reports true if mouse button is pressed.
	Reports true if specified key is pressed. Useful for controlling moving objects, such as in games.
	Reports distance from the specified sprite or mouse pointer. Useful in projects that require precision sensing and movement.
	Sets the timer to zero. Handy for when a project or new game level is started.
	Reports the value of the timer in seconds. (The timer is always running.)
	Reports a property or variable of another sprite. Select from: x-position, y-position, direction, costume #, size, and volume. Aids connectivity between sprites in a project.
	Reports the volume (from 1 to 100) of sounds detected by the computer microphone. More precise than **loud?**, it can be used to make sprites react to a certain voice level.
	Reports true if computer microphone detects a sound volume greater than 30 (on scale of 1 to 100).

Reports the value of specified sensor, such as one of the Pi's GPIO pins (or via a connected PicoBoard or LEGO WeDo).

Reports true if specified sensor is pressed. Only used with a connected PicoBoard.

Operators

These provide various mathematical and Boolean operations, along with functions for handling strings.

Adds two numbers.

Subtracts second number from first number.

Multiplies two numbers.

Divides first number by second number.

Picks a random integer within the specified range.

Reports true if first value is less than second.

Reports true if two values are equal.

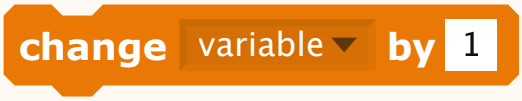
	Reports true if first value is greater than second.
	Reports true if both conditions are true.
	Reports true if either condition is true.
	Reports true if condition is false; reports false if condition is true.
	Concatenates (combines) the two strings.
	Reports the letter at the specified position in a string.
	Reports the number of letters in a string.
	Reports remainder from division of first number by second number.
	Reports closest integer to a number.
	Reports result of selected function (abs, sqrt, sin, cos, tan, asin, acos, atan, ln, log, e^, or 10^) applied to specified number.

Variables

These blocks only appear in the palette once a new variable (changeable value) or list (containing multiple items) is created.

Reports value of the variable. Each created variable has one of these blocks. Tick its box to show it on the stage. Creating a variable named 'AddOn' enables the use of Raspberry Pi add-on boards (see **magpi.cc/1TYX7Jg**).

Sets variable to specified value. Useful for resetting it at the start of a project. This can also be used to set the AddOn variable to use add-on boards such as the Explorer HAT, Pibrella, PiFace, PiGlow, and Sense HAT.



Changes selected variable by specified amount. Uses include altering the speed of an object, level number, or game score.

Shows the selected variable's monitor on the stage.

Hides the selected variable's monitor so it is not visible on the stage.

Reports all the items in the list. (The items are separated by spaces. However, if the items are individual letters or digits, spaces are omitted.)

Adds the specified item to the end of the list. The item can be a number or a string of letters and other characters.

Deletes one or all items from a list. Choosing 'last' deletes the last item in the list. Choosing 'all' deletes everything from the list. Deleting decreases the length of the list.

Inserts an item at the specified position in the list. Choosing 'any' inserts at a random place in the list. Choosing 'last' adds the item to the end of the list. The length of the list increases by 1.

Replaces an item in the list with the specified value. Choosing 'any' replaces a random item in the list. The length of the list does not change.

Reports the item at the specified position in the list. Choosing 'any' reports a random item in the list.

Reports how many items are in the list.

Reports true if the list contains the specified item. Note that the item must match exactly to report true.