



Gurukul Technicals Institute

Computer education · Robotics · Science projects · Delhi

FREE EBOOK

ROBOTICS AND ELECTRONICS

Coding for Kids with Scratch

Make games and animations with Scratch blocks

INSIDE THIS BOOK

- Meet Scratch
- A tour of the editor
- Motion, looks and sound: the dancing cat
- Events and broadcasting: a talking story
- Loops and conditions: the maze game
- ...and 5 more chapters, practice questions and a glossary

Beginner

Children of class 3 to 8 and their teachers

engineergurukul.com

+91 97116 99511

Coding for Kids with Scratch

A friendly first coding book for children of class 3 to 8 and their teachers. You will learn the Scratch editor, sprites, events, loops, conditions, variables and broadcasting, then build eight projects step by step, from a dancing cat to a maze, a quiz, a piano and a pong game, and learn to fix bugs and share your work safely.

WHO IT IS FOR	Children of class 3 to 8 and their teachers
LEVEL	Beginner
SUBJECT AREA	Robotics and electronics
WHAT IS INSIDE	10 chapters, practice questions with answers, and a glossary

HOW TO USE IT

Read one chapter at a time and try the examples as you go: on a computer, at your workbench or in your notebook. The boxes give tips, safety points and short activities. When you finish, test yourself with the practice questions and check your answers at the end. Words you may not know are explained in the glossary.



Gurukul Technicals Institute

A-10, Chief Commissioner Colony, opposite Rana Pratap Bagh, Delhi 110007

Phone 011 2721 1512 · Mobile and WhatsApp +91 97116 99511

engineergurukul@yahoo.co.in · engineergurukul.com

Free ebook. You may read, print and share this ebook for learning. It may not be sold or changed.

© 2026 Gurukul Technicals Institute. First edition, October 2026. The information was checked when it was written; software, prices and rules change, so for anything official always check the latest source.

What you will find inside

1	Meet Scratch	4
2	A tour of the editor	5
3	Motion, looks and sound: the dancing cat	7
4	Events and broadcasting: a talking story	10
5	Loops and conditions: the maze game	12
6	Variables: catch the apple and a quiz	14
7	Drawing and music: the pen and the piano	16
8	Project 8: Simple pong	19
9	Debugging: finding and fixing mistakes	21
10	Sharing safely and what to try next	22
	Practice questions	24
	Answers	25
	Glossary	26

Meet Scratch

Imagine telling a cartoon cat to dance, talk and play a game, just by snapping coloured blocks together like toy bricks. That is **Scratch**. It is a free programming language made for young learners. A **programming language** is a way of giving instructions to a computer.

Scratch was created by a team at MIT, a famous university in the United States. Millions of children around the world use it to make games, stories and animations.

Why blocks?

In most programming languages, you type words and symbols. One missing bracket and the program stops. In Scratch, each instruction is a **block**. Blocks fit together only in ways that make sense, a bit like puzzle pieces. So you can spend your time on ideas, not on spelling.

The ideas you learn in Scratch are the same ideas used in real programming: steps in order, loops, decisions and variables. When you move to Python or JavaScript later, you will already know how to think like a programmer.

Where to use Scratch

Way	What you need	Good for
In a web browser	A computer with internet, at scratch.mit.edu	Using it anywhere, saving online, sharing
The Scratch app	The free app, downloaded from scratch.mit.edu	Working without internet

You do not need an account to make projects in the browser. An account lets you save your work online and share it. Ask a parent or teacher to help you make one.

DID YOU KNOW

Scratch is translated into many languages. Look for the globe icon in the menu bar at the top of the editor to change the language.

How this book shows blocks

This book writes Scratch blocks as text, so you can read and copy them. The words match the blocks in Scratch. Brackets show the parts you can change:

- `( )` round brackets hold a number or a word you type, such as `move (10) steps`.
- `[ ]` square brackets show a choice from a drop-down menu, such as `when [space] key pressed`.
- `< >` pointed brackets hold a yes-or-no question, such as `<touching [Bowl]?>`.
- A round block that reports a value, such as `(score)`, `(answer)` or `(x position)`, is also shown in round brackets. You drag it into a slot.
- Blocks indented under `forever`, `repeat` or `if` go inside that block.
- `end` shows where a `forever`, `repeat` or `if` block closes. You do not drag an `end` block. It is just the bottom of the C-shaped block.

REMEMBER

Scratch is free, made at MIT, and runs in a browser or as an app. You build programs by snapping blocks together.

CHAPTER 2

A tour of the editor

Open Scratch and click **Create**. You will see the **editor**, the screen where you build projects. It has four main areas.

Area	Where it is	What it does
Block palette	Left side	All the blocks, sorted into coloured groups

Area	Where it is	What it does
Code area	Middle	Where you drag blocks to build scripts
Stage	Top right	Where your project plays
Sprite pane	Below the stage	Shows your sprites, and the stage's backdrops at the far right

Sprites and the stage

A **sprite** is a character or object in your project. The orange cat you see at the start is a sprite. A project can have many sprites: a dog, a ball, a car. Each sprite has its own scripts.

The **stage** is the background where sprites move. The picture on the stage is called a **backdrop**. You can have several backdrops and switch between them, like scenes in a play.

A **script** is a stack of blocks joined together. It runs from top to bottom.

Adding sprites and backdrops

- To add a sprite, point to the round cat-face button at the bottom right of the sprite pane and click **Choose a Sprite**. You can also paint your own.
- To add a backdrop, use the similar button at the far right, below the stage, and click **Choose a Backdrop**.
- To change how a sprite looks, open the **Costumes** tab at the top left. Sounds are under the **Sounds** tab.

Where things are on the stage

The stage is like a graph. Every point has two numbers.

- **x** tells you how far left or right. It goes from -240 at the left edge to 240 at the right edge.
- **y** tells you how far up or down. It goes from -180 at the bottom to 180 at the top.
- The centre of the stage is x: 0, y: 0.

Directions are measured in degrees. Direction 90 points right, -90 points left, 0 points up and 180 points down.

TRY THIS

Drag the cat around the stage. Watch the x and y numbers in the sprite pane change. Can you find the top-left corner? It is near x: -240, y: 180.

Your very first script

Before the big projects, try this warm-up.

1. Click the blue **Motion** circle in the block palette.
2. Drag `move (10) steps` into the code area.
3. Click the block. The cat moves a little to the right.
4. Click the 10 inside the block and type 50. Click the block again. The cat moves further.
5. Find the turn block whose curved arrow goes the same way as a clock's hands. This book calls it `turn right (15) degrees`. Drag it under the move block until it snaps on. Click the stack. The cat moves, then turns.

To remove a block, drag it back to the palette. To tidy a messy code area, right-click an empty part of it and choose **Clean up Blocks**.

Starting and stopping

The **green flag** above the stage starts your project. The **red stop sign** stops everything. Most projects begin with the block `when green flag clicked`.

TIP

Save often. In the browser, if you are signed in, use **File** and then **Save now**. Without an account, use **File** and then **Save to your computer**. This makes a file ending in .sb3 that you can open again later with **Load from your computer**.

CHAPTER 3

Motion, looks and sound: the dancing cat

Three groups of blocks bring a sprite to life.

Group	Colour	What its blocks do	Example block
Motion	Blue	Move and turn the sprite	<code>move (10) steps</code>
Looks	Purple	Change costume, size and colour; make it talk	<code>say (Hello!) for (2) seconds</code>
Sound	Pink	Play sounds and change volume	<code>start sound [Meow]</code>

A **costume** is one picture of a sprite. The cat has two costumes, with its legs in different places. Switching quickly between them makes the cat look as if it is walking.

A few more blocks you will use often:

- `say` shows a speech bubble. `think` shows a thought bubble, like in a comic.
- `hide` makes a sprite disappear, and `show` brings it back.
- `set size to (100) %` makes a sprite bigger or smaller. 50 is half size, 200 is double.
- You can record your own voice. Open the **Sounds** tab, point to the button at the bottom left and choose **Record**. Your computer needs a microphone, and the browser will ask your permission.

Project 1: The dancing cat

Goal: the cat dances from side to side, changes colour and meows.

1. Start a new project. The cat is already there.
2. Choose a backdrop you like.
3. Click the cat in the sprite pane so its scripts show.
4. Build this script:

```
when green flag clicked
go to x: (0) y: (0)
repeat (10)
  move (30) steps
  next costume
  wait (0.3) seconds
  move (-30) steps
  next costume
  wait (0.3) seconds
end
change [color] effect by (25)
start sound [Meow]
say (Did you like my dance?) for (2) seconds
```

1. Click the green flag and watch the cat dance.

How it works

- `go to x: (0) y: (0)` puts the cat in the centre at the start, so it does not wander off.
- `repeat (10)` runs the blocks inside it ten times.
- `move (30) steps` moves right, and `move (-30) steps` moves back. A minus sign means the opposite way.
- `next costume` swaps the cat's picture, so its legs move.
- `change [color] effect by (25)` gives the cat a new colour.

EXAMPLE

Aarohi in class 4 in Nagpur changed the `repeat (10)` to `repeat (20)` and the waits to 0.1 seconds. Her cat danced so fast that her little brother laughed until he fell off his chair.

TRY THIS

Add a second script to the cat: `when this sprite clicked`, then `play sound [Meow] until done` and `turn right (15) degrees`. Click the cat while it dances.

Events and broadcasting: a talking story

An **event** is something that happens, such as a click, a key press or the green flag. The yellow **Events** blocks wait for something to happen, then start a script. They are called **hat blocks** because they sit on top of a script, like a hat.

Event block	When the script starts
when green flag clicked	When you click the green flag
when [space] key pressed	When you press that key
when this sprite clicked	When you click the sprite
when backdrop switches to []	When the stage changes to that backdrop
when I receive []	When a broadcast message arrives

What is broadcasting?

Each sprite runs its own scripts. To make sprites work together, one sprite can send a **message** to all the others. This is called **broadcasting**. Any sprite with a matching **when I receive** block starts its script when the message arrives.

Think of the school bell. When it rings, every class knows it is break time. The bell does not tell each class separately. It broadcasts.

Project 2: The talking story

Goal: two characters talk to each other, and the scene changes.

1. Keep the cat. Add a second sprite, such as a dog or a parrot, from the **Animals** group.
2. Choose two backdrops, for example a park and a beach.
3. Drag the cat to the left and the second sprite to the right.

Script for the cat:

```
when green flag clicked
switch backdrop to [park]
say (Hi! I am Meenu. Who are you?) for (3) seconds
broadcast [your turn]
```

Script for the second sprite:

```
when I receive [your turn]
say (I am Moti. Shall we go to the beach?) for (3) seconds
broadcast [go to beach]
```

Script for the cat again:

```
when I receive [go to beach]
switch backdrop to [beach]
say (Yay! I love the sea!) for (2) seconds
```

To make a new message, click the drop-down on a **broadcast** block and choose **New message**. Type a name that explains what it is for.

TIP

Your backdrop names may be different from "park" and "beach". Use the names shown in the drop-down menu.

REMEMBER

One sprite broadcasts a message. Every sprite with a matching **when I receive** block starts at once.

TRY THIS

Add a third character who appears only at the beach. Give it **when backdrop switches to [beach]**, then show and say. Use **hide under when green flag clicked** so it is hidden at the start.

Loops and conditions: the maze game

Loops

A **loop** repeats blocks. Scratch has three loop blocks in the orange **Control** group.

Loop block	What it does
<code>repeat (10)</code>	Repeats a set number of times
<code>forever</code>	Repeats until you click stop
<code>repeat until < ></code>	Repeats until a question becomes true

Conditions

A **condition** is a question with a yes or no answer, such as "Is the sprite touching the wall?" The `if < > then` block runs the blocks inside it only if the answer is yes. The `if < > then else` block does one thing for yes and another for no.

Conditions often use the light blue **Sensing** blocks, like `<touching [mouse-pointer]?>` and `<key [space] pressed?>`.

Repeat until: stop when something happens

`repeat until` keeps going until its question becomes true. This script makes the cat walk until it reaches the edge of the stage:

```
when green flag clicked
go to x: (-200) y: (0)
point in direction (90)
repeat until <touching [edge]?>
  move (5) steps
  next costume
end
say (I reached the edge!) for (2) seconds
```

The blocks after the loop run only when the loop has finished.

Project 3: The maze game

Goal: guide a ball through a maze with the arrow keys, without touching the walls, to reach a green goal.

Make the maze:

1. Click the stage's thumbnail at the far right of the sprite pane, then open the **Backdrops** tab. You will see the blank white backdrop.
2. Use the rectangle tool to draw thick blue walls on it, with a path between them.
3. Draw a small green square at the end of the path. This is the goal.

Make the player:

1. Add the **Ball** sprite, or paint a small circle.
2. In the sprite pane, set its size to about 30 so it fits the path.
3. Note the x and y of the start of your path. You will use them below.

```
when green flag clicked
go to x: (-200) y: (150)
forever
  if <key [right arrow] pressed?> then
    change x by (3)
  end
  if <key [left arrow] pressed?> then
    change x by (-3)
  end
  if <key [up arrow] pressed?> then
    change y by (3)
  end
  if <key [down arrow] pressed?> then
    change y by (-3)
  end
  if <touching color [blue]?> then
    go to x: (-200) y: (150)
  end
  if <touching color [green]?> then
    say (You win!) for (2) seconds
    stop [all]
  end
end
```

To set the colour in **touching color**, click the colour box and use the dropper tool to click on your wall. Change -200 and 150 to the start position of your own maze.

EXAMPLE

Faizan in class 6 in Hyderabad made his walls too thin. The ball jumped right over them. He made the walls thicker and changed the steps from 3 to 2, and the game worked.

DID YOU KNOW

The `forever` loop in this game checks the keys and the walls many times every second. That is why the ball reacts at once when you press a key.

CHAPTER 6

Variables: catch the apple and a quiz

A **variable** is a box with a name that holds a value, such as a score. The value can change while the program runs. To make one, go to the dark orange **Variables** group and click **Make a Variable**. Give it a clear name, such as `score`.

Block	What it does
<code>set [score] to (0)</code>	Puts a value in the box
<code>change [score] by (1)</code>	Adds to the value; use -1 to take away
<code>show variable [score]</code>	Shows the box on the stage

Project 4: Catch the apple

Goal: apples fall from the sky. Move a bowl left and right to catch them. Each catch adds one point.

1. Delete the cat: right-click it in the sprite pane and choose **delete**.
2. Add the **Apple** sprite and the **Bowl** sprite from the library.
3. Make a variable called `score` for all sprites.

Script for the Bowl:

```

when green flag clicked
go to x: (0) y: (-150)
forever
  if <key [right arrow] pressed?> then
    change x by (10)
  end
  if <key [left arrow] pressed?> then
    change x by (-10)
  end
end
end

```

Script for the Apple:

```

when green flag clicked
set [score] to (0)
go to x: (pick random (-220) to (220)) y: (170)
forever
  change y by (-5)
  if <touching [Bowl]?> then
    change [score] by (1)
    go to x: (pick random (-220) to (220)) y: (170)
  end
  if <(y position) < (-170)> then
    go to x: (pick random (-220) to (220)) y: (170)
  end
end
end

```

`pick random (-220) to (220)` chooses a new number each time, so the apple falls from a different place. `y position` is a round Motion block that tells you how high the apple is.

TRY THIS

Make a second variable called `missed`. Add 1 to it each time the apple reaches the bottom. When `missed` reaches 3, say "Game over" and stop `[all]`.

Project 5: The quiz game

Goal: the cat asks questions, checks the answers and keeps a score.

The `ask ( )` and `wait` block in **Sensing** shows a question and a box to type in. Whatever the player types is stored in the round `answer` block.

```

when green flag clicked
set [score] to (0)
ask (What is 7 x 8?) and wait
if <(answer) = (56)> then
    say (Correct!) for (2) seconds
    change [score] by (1)
else
    say (Oops! It is 56.) for (2) seconds
end
ask (What is the capital of India?) and wait
if <(answer) = (New Delhi)> then
    say (Correct!) for (2) seconds
    change [score] by (1)
else
    say (Oops! It is New Delhi.) for (2) seconds
end
say (join (Your score is ) (score)) for (3) seconds

```

The **join** block in the green **Operators** group sticks two pieces of text together. Leave a space after "is" inside the first box, or the words will run together.

When Scratch compares words with **=**, it does not mind capital or small letters, so "new delhi" also counts as correct. But spelling and spaces must match.

EXAMPLE

Rohan, a class 7 student in Varanasi, made a 10-question quiz on his science chapter for his class. He added a variable for the question number and a sound for each right answer. His teacher used it for revision.

REMEMBER

A variable is a named box that holds a value. Set it at the start, change it while the game runs, and show it on the stage.

CHAPTER 7

Drawing and music: the pen and the piano

Some blocks are not shown at first. They live in **extensions**. Click the **Add Extension** button at the bottom-left corner of the editor and choose one. Its blocks appear as a new group at the bottom of the palette.

Project 6: Drawing with the pen

Add the **Pen** extension. A sprite with its pen down draws a line wherever it moves, like a pencil.

```
when green flag clicked
erase all
pen up
go to x: (0) y: (0)
point in direction (90)
set pen color to [red]
set pen size to (3)
pen down
repeat (4)
  move (100) steps
  turn right (90) degrees
end
pen up
```

This draws a square. Each corner turns 90 degrees, and $4 \times 90 = 360$, one full turn.

Change the shape:

Shape	Repeat	Turn each time
Triangle	3	120 degrees
Square	4	90 degrees
Hexagon	6	60 degrees
Five-pointed star	5	144 degrees

For any regular shape, the turn is 360 divided by the number of sides. The star is different: its sprite turns round twice while drawing it, so the turns add up to $5 \times 144 = 720$ degrees.

A rainbow spiral: make a variable called `length`. Then try this:

```

when green flag clicked
erase all
go to x: (0) y: (0)
set [length] to (5)
pen down
repeat (60)
  move (length) steps
  turn right (91) degrees
  change pen [color] by (5)
  change [length] by (4)
end
pen up

```

Project 7: The piano

Add the **Music** extension. The block `play note (60) for (0.5) beats` plays a note. Note 60 is middle C. Each step up the white keys of a piano is a higher number.

Indian name (Sa = C)	Western name	Note number	Key to press
Sa	C	60	a
Re	D	62	s
Ga	E	64	d
Ma	F	65	f
Pa	G	67	g
Dha	A	69	h
Ni	B	71	j
Upper Sa	C	72	k

Make one small script for each key, like this:

```

when [a] key pressed
play note (60) for (0.5) beats

```

```

when [s] key pressed
play note (62) for (0.5) beats

```

Repeat for the other six keys. Now type a, s, d, f, g, h, j, k to play Sa Re Ga Ma Pa Dha Ni Sa.

TRY THIS

Play "Twinkle, twinkle, little star" with these keys: a a g g h h g, then f f d d s s a.

DID YOU KNOW

You can paint eight white rectangles as sprites and give each one when this sprite clicked with its own note. Then you can play your piano with the mouse, or on a touch screen.

CHAPTER 8

Project 8: Simple pong

Pong is one of the oldest video games. A ball bounces around, and you move a paddle to stop it falling off the bottom. This project uses almost everything you have learnt.

Set up

1. Start a new project and delete the cat.
2. Add the **Ball** sprite and the **Paddle** sprite.
3. Paint the backdrop with a thick red band, about as thick as your finger on the screen, along the bottom edge. Missing the ball means touching the red band.
4. Make a variable called `score`.

Script for the Paddle

The paddle follows the mouse left and right, but stays near the bottom.

```
when green flag clicked
go to x: (0) y: (-140)
forever
  set x to (mouse x)
end
```

Script for the Ball

```
when green flag clicked
set [score] to (0)
go to x: (0) y: (100)
point in direction (pick random (135) to (225))
forever
  move (6) steps
  if on edge, bounce
  if <touching [Paddle]?> then
    point in direction ((180) - (direction))
    move (10) steps
    change [score] by (1)
  end
  if <touching color [red]?> then
    say (Game over!) for (2) seconds
    stop [all]
  end
end
end
```

How it works

- `point in direction (pick random (135) to (225))` sends the ball downwards at a random angle, so each game is different.
- `if on edge, bounce` makes the ball bounce off the walls and the top.
- When the ball touches the paddle, `((180) - (direction))` flips its direction from downwards to upwards, like a real bounce.
- `move (10) steps` straight after the bounce moves the ball away from the paddle, so it does not touch it again and get stuck.
- The `-` block is in the green **Operators** group. Put the round `direction` block from **Motion** in its second slot.

EXAMPLE

Simran and Joseph, class 8 friends in Amritsar, added a rule: every 5 points, the ball moves 1 step faster. They used a variable called `speed` in place of the 6 in `move (6) steps`, and changed it by 1 whenever the score could be divided by 5.

TIP

If the game is too hard, make the paddle wider in the **Costumes** tab, or lower the ball's steps from 6 to 4.

TRY THIS

Add a sound when the ball hits the paddle. Then add a second level with a new backdrop when the score reaches 10.

CHAPTER 9

Debugging: finding and fixing mistakes

A mistake in a program is called a **bug**. Finding and fixing it is called **debugging**. Every programmer, from a class 3 student to a professional, writes bugs. Good programmers are not people who never make mistakes. They are people who are good at finding them.

Common bugs in Scratch

What happens	Likely bug	Fix
Nothing happens on the green flag	The script has no <code>when green flag clicked</code> on top, or the block is not joined	Drag the hat block onto the top until it snaps
The script works for the wrong sprite	The code was built on another sprite	Click the right sprite before adding code
The sprite starts in a strange place	No starting position	Add <code>go to x: () y: ()</code> at the start
The score does not reset	No <code>set [score] to (0)</code> at the start	Add it under the green flag
Things happen too fast to see	No waits inside a loop	Add <code>wait (0.5) seconds</code>
<code>touching color</code> never works	The colour is slightly different	Use the dropper to pick the exact colour

Debugging steps

1. Say what should happen, then watch what really happens. Where do they differ?

2. **Click one script at a time.** Clicking a script runs it alone. A running script glows yellow.
3. **Show your variables.** Tick the box next to a variable so its value appears on the stage.
4. **Slow it down.** Add a `wait (1) seconds` so you can see each step.
5. **Use say blocks.** Put `say (score)` or `say (direction)` in the loop to see what the computer knows.
6. **Change one thing at a time,** and test after each change.

EXAMPLE

Neha in class 5 in Ranchi could not understand why her apple never fell. The apple's script was on the Bowl sprite by mistake. So the bowl was falling off the screen while she watched the apple. She dragged the script onto the Apple sprite in the sprite pane, which copies it, and deleted the old one.

Before a big change to a project that works, save a copy with a new name, such as "maze version 2". If the change goes wrong, you can go back to the copy.

TIP

Explain your code out loud, block by block, to a friend or even to a soft toy. Very often you spot the bug while you are explaining it.

CHAPTER 10

Sharing safely and what to try next

Sharing your projects

When you are signed in, the **Share** button puts your project on the Scratch website. Anyone in the world can then see it, play it, comment on it and **remix** it. To remix means to make a copy of someone else's project and change it. Scratch shows who made the original, so they get credit.

Staying safe online

Scratch has rules for its community, and a team that checks for problems. You also have a part to play.

- **Keep personal details private.** Never share your full name, school, address, phone number or passwords. Do not post photos of yourself.
- **Choose a username that is not your real name.**
- **Be kind.** Leave helpful comments. If you would not say it face to face, do not type it.
- **Report problems.** If you see something unkind or worrying, use the **Report** button and tell a parent or teacher.
- **Do not share other people's photos, music or work** as if they were your own.

SAFETY

If anyone online asks you for personal details, wants to chat privately or asks to meet you, do not reply. Tell a parent or teacher at once.

REMEMBER

Share your projects, not your personal details. Be kind, give credit when you remix, and tell an adult if something feels wrong.

Ideas for more projects

- **An animated greeting card** for Diwali, Eid, Christmas or a friend's birthday.
- **A times-table trainer** that asks random questions using `pick random`.
- **A racing game** with a car that follows a road drawn on the backdrop.
- **A story with choices**, where the player decides what happens next.
- **A science animation**, such as the water cycle or the phases of the Moon.
- **Clones:** in **Control**, `create clone of [myself]` makes copies of a sprite. Use it to make many falling apples at once.
- **My Blocks:** in the pink **My Blocks** group, you can make your own block, such as "draw square", and use it again and again.

DID YOU KNOW

For younger brothers and sisters of about five to seven, there is a free tablet app called ScratchJr. It uses picture blocks instead of words.

Most of all, keep making things. Every project teaches you something new.

CHECK YOURSELF

Practice questions

1. What is Scratch, and who created it?
2. Name two ways to use Scratch.
3. What is a sprite?
4. What is the stage, and what is a backdrop?
5. What are the x and y values at the centre of the stage?
6. Which direction does a sprite face when its direction is 90?
7. What is a costume, and how can costumes make a sprite look as if it is walking?
8. Which colour are the Motion blocks, and what do they do?
9. Why are event blocks called hat blocks?
10. What does broadcasting do?
11. What is the difference between `repeat (10)` and `forever`?
12. What is a condition? Give an example.
13. In the maze game, what happens when the ball touches a blue wall?
14. What is a variable?
15. In catch the apple, why do we use `pick random (-220) to (220)`?
16. In the quiz game, where is the player's typed reply stored?
17. Which block sticks two pieces of text together?
18. How do you add the Pen or Music blocks to the palette?

19. How many times should you repeat, and how many degrees should you turn, to draw a hexagon?
20. In the piano project, which note number is middle C?
21. In pong, why does the ball move 10 steps straight after touching the paddle?
22. What is a bug, and what is debugging?
23. Your score keeps the old value when you start a new game. What is the likely bug?
24. Give three personal details you should never share online.
25. What does it mean to remix a project?

ANSWER KEY

Answers

1. A free block-based programming language for young learners, created by a team at MIT.
2. In a web browser at scratch.mit.edu, or with the free Scratch app downloaded from there.
3. A character or object in a Scratch project, with its own scripts.
4. The stage is the area where the project plays. A backdrop is the picture on the stage.
5. x: 0 and y: 0.
6. Right.
7. A costume is one picture of a sprite. Switching quickly between costumes with legs in different places looks like walking.
8. Blue. They move and turn the sprite.
9. They sit on top of a script, like a hat, and start it when something happens.
10. It sends a message to all sprites. Any sprite with a matching `when I receive` block starts its script.
11. `repeat (10)` runs the blocks ten times and then moves on. `forever` keeps running until the project is stopped.
12. A question with a yes or no answer, such as `<touching color [blue]?>`.
13. It goes back to the start position.
14. A named box that holds a value, such as a score, which can change while the program runs.
15. So the apple falls from a different place each time.

16. In the round `answer` block.
17. The `join` block in the Operators group.
18. Click the Add Extension button at the bottom-left corner and choose Pen or Music.
19. Repeat 6 times and turn 60 degrees each time.
20. 60.
21. To move it away from the paddle, so it does not touch it again and get stuck.
22. A bug is a mistake in a program. Debugging is finding and fixing it.
23. There is no `set [score] to (0)` under the green flag.
24. Any three: full name, school, address, phone number, passwords, photos of yourself.
25. To make a copy of someone else's project and change it, with credit to the original maker.

WORD LIST

Glossary

Backdrop

a picture shown on the stage, behind the sprites.

Block

one instruction in Scratch, shaped to fit with other blocks.

Broadcast

a message sent to all sprites to start their scripts.

Bug

a mistake in a program.

Condition

a question with a yes or no answer that decides what a program does.

Costume

one picture of a sprite; a sprite can have many.

Debugging

finding and fixing bugs.

Editor

the Scratch screen where you build projects.

Event

something that happens, such as a click or a key press, that can start a script.

Extension

an extra group of blocks you can add, such as Pen or Music.

Hat block

a block with a rounded top that starts a script when an event happens.

Loop

blocks that repeat other blocks.

Operator

a block that does maths, compares values or joins text.

Programming language

a way of giving instructions to a computer.

Remix

a changed copy of someone else's shared project, with credit to them.

Script

a stack of joined blocks that runs from top to bottom.

Sprite

a character or object in a Scratch project.

Stage

the area where a Scratch project plays.

Variable

a named box that holds a value which can change.

x and y

the two numbers that give a position on the stage, across and up.



Gurukul Technicals Institute

Computer education institute in Delhi, with authorised member centres, robotics classes and science projects, working models and DIY kits.

COURSES

Computer & Office Skills · Accounting & Business Software · Programming & Web Development · Design & Digital Skills · Hardware & Networking · Robotics & Coding · Other Programmes

PROJECTS AND KITS

Ready-made school projects · DIY and STEM kits · Robotics and coding · Components and spares · Engineering projects

CONTACT

Head office: A-10, Chief Commissioner Colony, opposite Rana Pratap Bagh, Delhi 110007

Phone: 011 2721 1512 · **Mobile and WhatsApp:** +91 97116 99511

Email: engineergurukul@yahoo.co.in

Website: engineergurukul.com · **Instagram:** [@science_projects_club](https://www.instagram.com/science_projects_club)

More free ebooks for computer courses, robotics and science projects: engineergurukul.com/resources/ebooks/

Free to read, print and share. Not for sale.