



# Coding for STEM:

## Simulating falling objects

MAV'24

 Dec 6, 2024 (Session H11 - 3.10pm) 

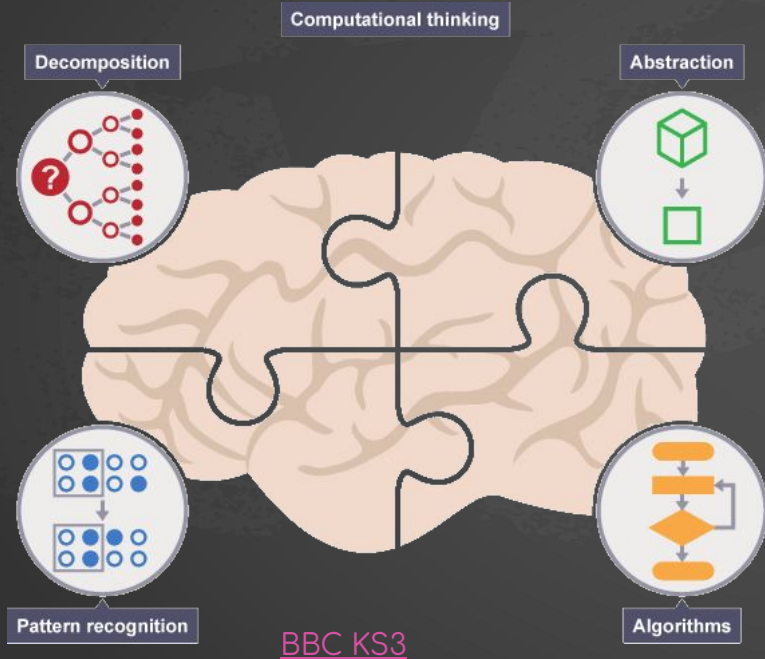


THE UNIVERSITY OF  
MELBOURNE

Dr. Max Stephens - University of Melbourne  
Prof. Sebastian Sardiña - RMIT University



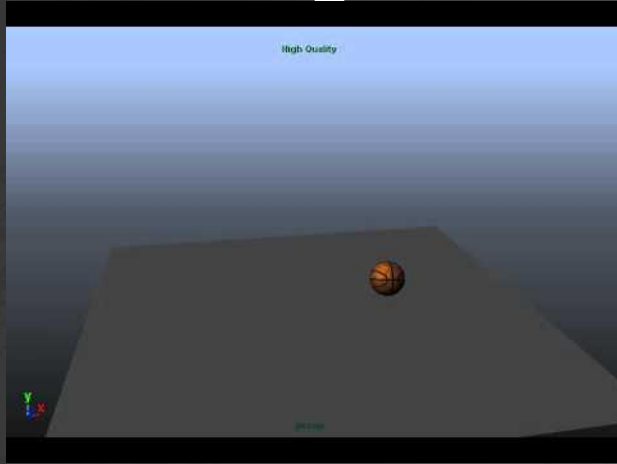
# The four cornerstones of Computational Thinking



- **Decomposition**: Break down complex problems; prevent from becoming overwhelmed.
- **Abstraction**: Strip away unnecessary details to see core features.
- **Pattern recognition**: find similarities, differences, trends, repetitions.
- **Algorithms**: step-by-step process to solve a problem or task.

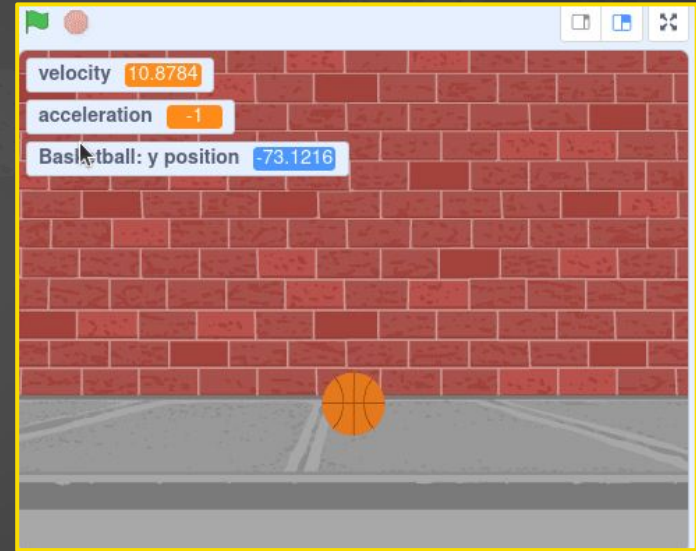
Also builds and support: *confidence* (lots of “aha” moments!), *tenacity*, *communication skills*, *curiosity*, *intentional attitude*, *growth mindset*.

# Falling objects..



# Break task down into key steps

1. Set-up the stage by placing ball (and background).
2. Drop ball at constant speed.
3. Hit floor and stop.
4. Added acceleration due to gravity.
5. Bounce back when hitting the floor.
6. Loss of energy when bouncing.
7. Extensions.



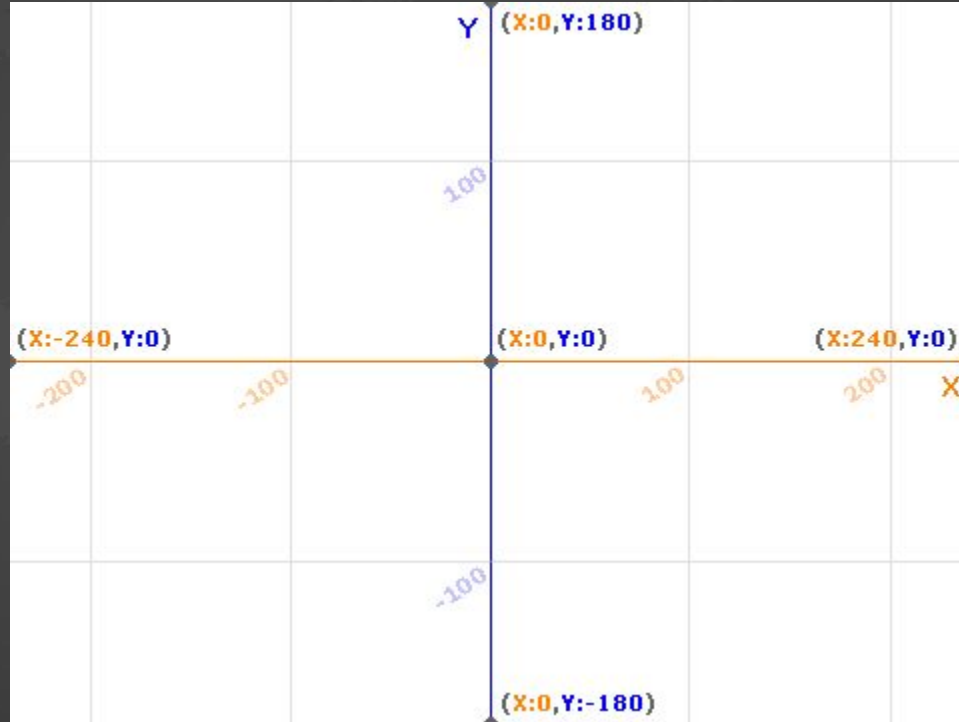
This tutorial is not a sample lesson, but a list of meaningful self-contained steps that constitutes the STEM coding **horizon** for the teacher to guide implementation. Each step may take multiple sessions.

**Specific timeline** will depend on: year level, experience with coding and Scratch, familiarity with STEM and mathematical ideas (e.g., variables and equations).

# Mathematical components promoted

1. Cartesian plane & coordinates (to place objects) - [VC2M6N01](#)
2. Positive & negative numbers (to represent direction, position).
3. Basic math operations (addition, multiplication, etc) - [VC2M4N06](#).
4. Number comparison (e.g., greater than) - [VC2M5N01](#)
5. Boolean logical expressions (e.g., or, and).
6. Use of variables for modeling - [VC2M4A01](#).
7. Proportion/percentage (to implement loss of energy) - [VC2M5N04](#).
8. Decimal numbers - [VC2M5N01](#).
9. Multiplying by -1 (to implement reversing of direction).

# Scratch Coordinate system





# Coding in Scratch

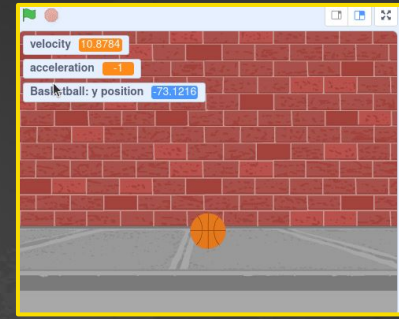
The image displays the Scratch programming environment. The main workspace contains a script for a basketball game. The script starts with a 'when green flag clicked' event, followed by setting 'velocity' to 0, 'acceleration' to -1, and 'ball\_y' to 150. It then moves the ball to x: 0 and y: ball\_y. A 'forever' loop follows, containing a 'set ball\_y to ball\_y + velocity' block, an 'if' condition 'ball\_y < -85 or ball\_y = -85', and a 'then' block that sets y to -85, starts a 'basketball bounce' sound, and sets velocity to velocity \* -1. The 'else' block sets y to ball\_y, turns the ball 2 degrees, and sets velocity to velocity + acceleration.

Four yellow callout boxes identify key components:

- Program**: Points to the script area.
- Stage**: Points to the stage area.
- Statement palette**: Points to the 'Motion' block category in the left sidebar.
- Sprites**: Points to the 'Basketball' sprite in the 'Sprites' panel.

The 'Sprites' panel shows a 'Basketball' sprite with x: 0, y: 135, size: 100, and direction: 32. The 'Stage' panel shows a brick wall backdrop and a 'Basketball' sprite.

# Two ways to use the project



## 1. Create the code incrementally.

- More demanding.
- Several sessions depending on existing knowledge and skills.
- Potentially higher-level of achievement.
- Step-by-step tutorial provided here: <https://bit.ly/3VcMNGx>

## 2. Remix/modify existing code.

- Full code provided by the teacher.
- Students first understand the code.
- Then, student modify code to achieve various objectives.
  - Different ball location.
  - Faster fall.
  - Less/more energy loss at bouncing.
  - Further abstraction (introduce new variables).
  - Fix existing bugs (maybe introduced by teacher in original code)
- After re-mixing, students may create their code from zero.



# Resources from today's session

- Tutorial: <https://bit.ly/3VcMNGx>
- Final program:
  - <https://scratch.mit.edu/projects/770606600/>
- Scratch: <https://scratch.mit.edu/>
- Similar project: <https://bit.ly/3GPrKn2>



## Falling Ball Tutorial @ Scratch MAV'24 - Coding for STEM Sebastian Sardina & Max Stephens

### Step-by-step project

- 1 - Setting up the stage
- 2 - Drop at constant speed
- 3 - Add backdrop
- 4 - Stop when hitting the ground
- 5 - Adding acceleration due to gravity
- 6 - Implementing bouncing
- 7 - Turning effect & bouncing sound
- 8 - Implementing friction & loss of energy
- 9 - Final program

2  
2  
4  
5  
6  
8  
9  
10  
11  
11

### Improvements & extensions

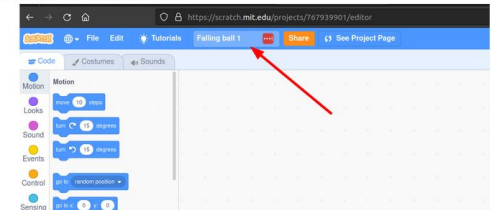
- Abstract floor level
- Factor out set-up as a procedure block
- Better stopping at floor (challenging!)
- Drag the ball anywhere

13  
13  
13  
13  
16

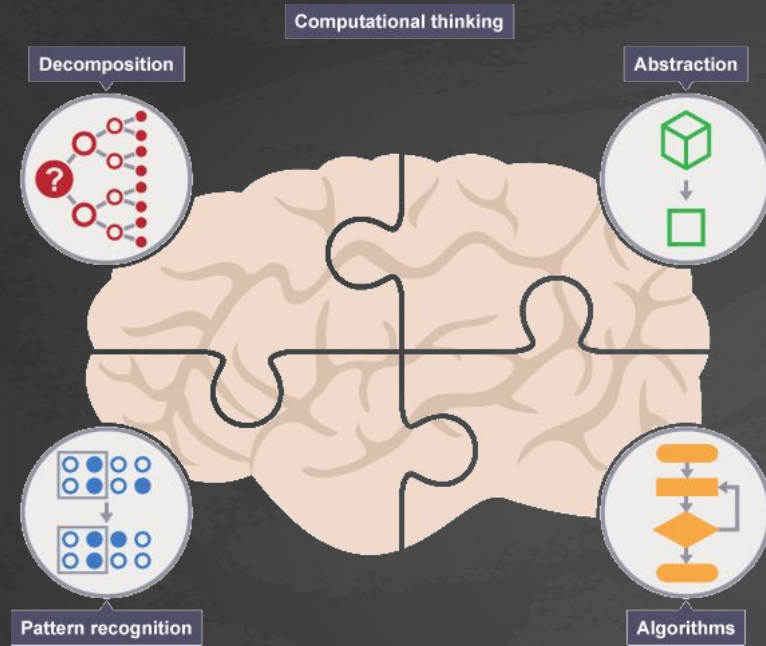
### Step-by-step project

#### 1 - Setting up the stage

First, create a Scratch Project and set its title:



Then, delete the default cat sprite in the Sprite tab:



Thank  
you!  
Q & A

Slides: <https://bit.ly/mav22-fball>

Questions? Comments? Suggestions? Share experiences?

Please contact us at:

- Sebastian Sardina: [sebastian.sardina@rmit.edu.au](mailto:sebastian.sardina@rmit.edu.au)
- Max Stephens: [m.stephens@unimelb.edu.au](mailto:m.stephens@unimelb.edu.au)

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**.

# Python



```
def main():
    floor = pygame.Rect((0, 670), (960, 80))

    ball_pos = (250, 50)
    ball_radius = 20
    ball_vel = (0, 0)
    ball_color = BLUE
    ball_accel = (0, GRAVITY)
    ball_energy_loss = 0.8

    running = True
    while running:
        # checks for mouse events
        for event in pygame.event.get():
            # app has been closed
            if event.type == QUIT:
                running = False
            # user releases mouse up - create the ball there!
            if event.type == pygame.MOUSEBUTTONUP:
                ball_pos = pygame.mouse.get_pos()

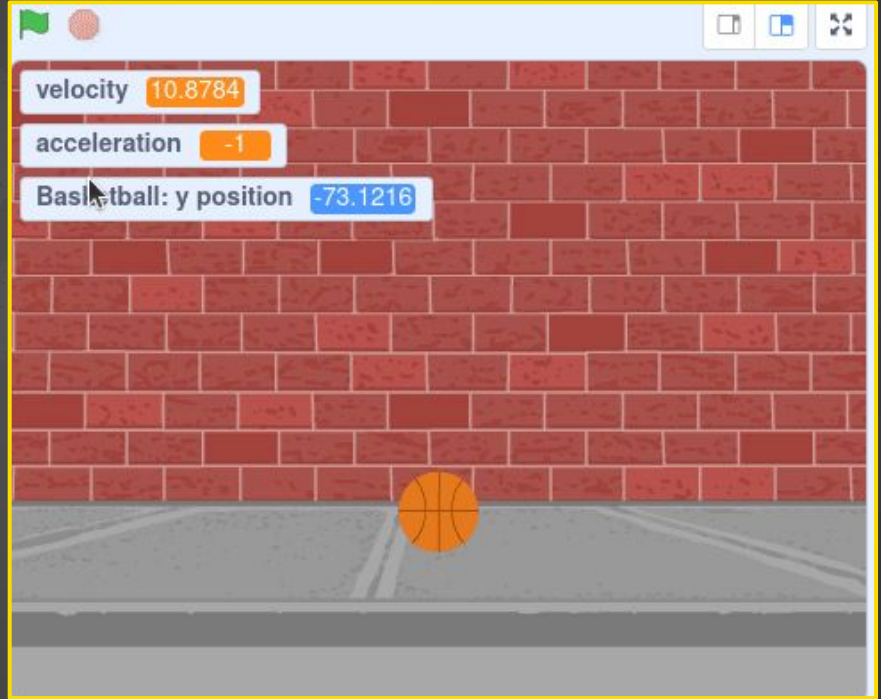
        # First update of ball position and velocity
        ball_vel = (ball_vel[0] + ball_accel[0], ball_vel[1] + ball_accel[1])
        ball_pos = (ball_pos[0] + ball_vel[0], ball_pos[1] + ball_vel[1])
        ball_hitbox = pygame.Rect((ball_pos[0]-ball_radius, ball_pos[1]-ball_radius), (ball_radius * 2, ball_radius * 2))

        # check for collision with floor; stick ball to floor if touching it
        if pygame.Rect.colliderect(ball_hitbox, floor):
            ball_pos = (ball_pos[0], math.ceil(floor.top - ball_radius)) # fix ball to the floor!
            ball_vel = (0, min([0, -(ball_energy_loss * ball_vel[1] + 0.5)]) )
            ball_hitbox = pygame.Rect((ball_pos[0]-ball_radius, ball_pos[1]-ball_radius), (ball_radius * 2, ball_radius * 2))

            if ball_vel[1] >= -1 and ball_vel[1] <= 0: # if vel upwards is very small, make it zero
                ball_vel = (ball_vel[0], 0)

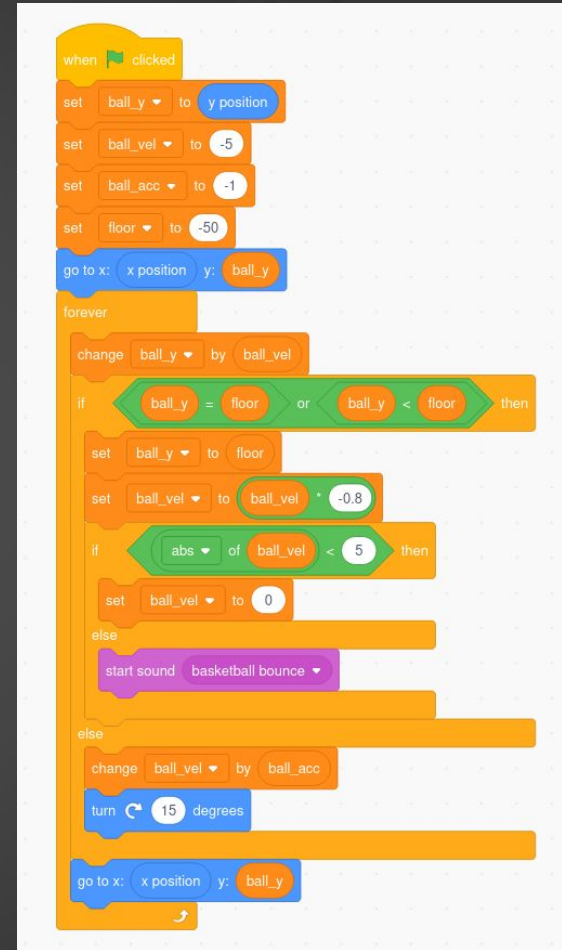
        # Now draw everything: background, floor, ball, hitbox
        SCREEN.fill(GREEN)
        pygame.draw.rect(SCREEN, GREY, floor)
        pygame.draw.circle(SCREEN, ball_color, ball_pos, ball_radius)
        # pygame.draw.rect(SCREEN, BLUE, ball_hitbox, 1)
```

# Step by step tutorial



# Steps

1. Set a ball in the upper-middle of the stage.
2. Naive (finte) fall.
3. Abstract ball's y-coordinate (variable bball\_y).
4. Constant free fall (forever).
5. Add a backdrop.
6. Collision with floor (if ball\_y = 80) - first try!
7. Collision with floor (if ball\_y = 80) - works!
8. Abstract velocity (variable ball\_acc).
9. Add acceleration (change ball\_vel via ball\_acc).
10. Perfect bouncing (multiplication by -1).
11. Add bouncing sound and rolling.
12. Model loss of energy at bounce (multiply by  $< -1$ ).
13. Stabilize at floor.
14. Allow any initial location.
15. Abstract acceleration & floor.





# 1 - Set a ball in the upper-middle

The image shows the Scratch project editor for a game titled "Fall Ball INCREMENTAL". The project is owned by "ssardina". The interface includes a top menu bar with options like Settings, File, Edit, Share, and See Project Page. Below this is a toolbar with tabs for Code, Costumes, and Sounds. The left sidebar contains a list of block categories: Motion, Looks, Sound, Events, Control, Sensing, Operators, Variables, and My Blocks. The main workspace is a grid where a script is being built. The script starts with a "when green flag clicked" event block, followed by a "go to x: 0 y: 150" block. A small basketball sprite is visible in the upper-middle of the stage. The right sidebar shows the "Stage" area with a "Basketball" sprite selected, and its properties: x: 0, y: 150, size: 100, direction: 90. Below the stage is a "Backdrops" area with one backdrop named "1".

Scratch Project: Fall Ball INCREMENTAL

Code Editor:

- Code** (selected)
- Costumes
- Sounds

Block Categories:

- Motion
- Looks
- Sound
- Events
- Control
- Sensing
- Operators
- Variables
- My Blocks

Script:

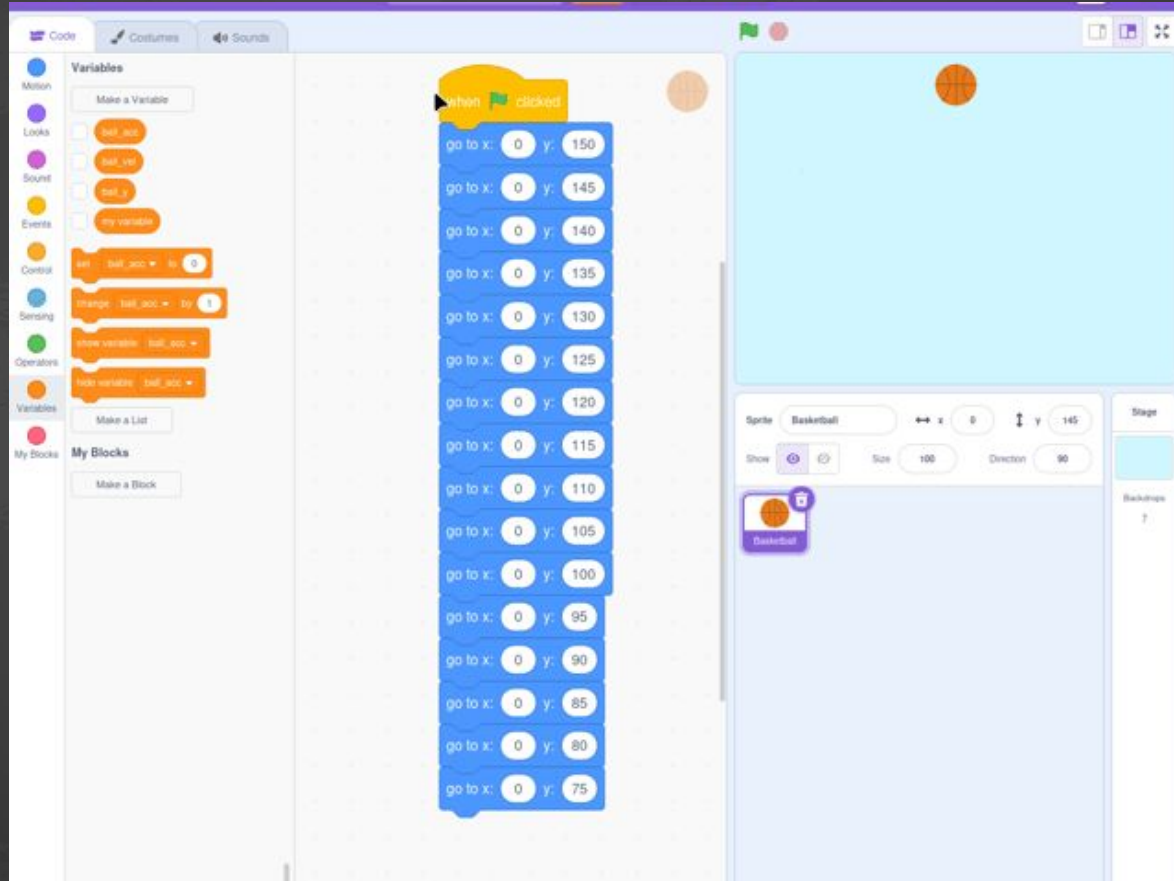
- when green flag clicked
- go to x: 0 y: 150

Stage:

- Sprite: Basketball
- x: 0, y: 150
- Show: ☒ ☐
- Size: 100
- Direction: 90
- Backdrops: 1



# 2 - Naive (finite) fall



# 3 - Abstract ball's y-coordinate

The image shows a Scratch project titled "Fall Ball INCREMENT". The interface includes a top menu bar with options like Settings, File, Edit, Share, and a toolbar with a green flag icon and a red circle icon. The left sidebar contains a "Variables" section with a "ball\_y" variable set to 150. The main workspace displays a basketball sprite and a code block that sets the ball's y-coordinate to 150 when the green flag is clicked. The bottom status bar shows the sprite is a "Basketball" and its coordinates are (0, 150).

Scratch

Settings File Edit Fall Ball INCREMENT Share See Project Page Tutorials Saving project...

Code Costumes Sounds

Variables

Make a Variable

☒ ball\_y ☐ my variable

set ball\_y to 0

change ball\_y by 1

show variable ball\_y

hide variable ball\_y

Make a List

My Blocks

Make a Block

when green flag clicked

set ball\_y to 150

go to x: 0 y: ball\_y

ball\_y 150

Sprite Basketball x 0 y 150

# 4 - Constant free fall

The image shows the Scratch project editor for a game titled "Fall Ball INCREMENTAL". The project is in the "Code" tab. The script for the "when green flag clicked" event is as follows:

```
when green flag clicked
  set ball_y to 150
  go to x: 0 y: ball_y
  forever loop
    change ball_y by -5
    go to x: 0 y: ball_y
```

The "Variables" panel on the left shows a variable named "ball\_y" of type "my variable". The "My Blocks" panel is empty. The "Stage" area on the right shows a basketball sprite at the top of the stage. The "Sprite" panel at the bottom right shows the "Basketball" sprite with x-coordinate -207, y-coordinate -96, size 100, and direction 90.

# 5 - Add a backdrop

The image shows the Scratch project editor for a game titled "Fall Ball INCREMENTAL". The interface is divided into several sections:

- Top Bar:** Includes the Scratch logo, settings, file, edit, and share buttons. The project name "Fall Ball INCREMENTAL" is displayed, along with a "Share" button and a "See Project Page" link. There are also links for "Tutorials", "Save Now", and a user profile for "ssardina".
- Left Panel:** Contains tabs for "Code", "Costumes", and "Sounds". Under the "Code" tab, there are sections for "Variables" and "My Blocks". The "Variables" section shows a variable named "ball\_y" with a value of 70. The "My Blocks" section shows a block named "Basketball".
- Code Area:** Displays a script for the "Basketball" sprite. The script starts with a "when green flag clicked" event, followed by a "set ball\_y to 150" block, a "go to x: 0 y: ball\_y" block, and a "forever" loop containing a "change ball\_y by -5" block and another "go to x: 0 y: ball\_y" block.
- Stage:** Shows a red brick wall backdrop. A basketball sprite is positioned in the center of the wall. The "ball\_y" variable is displayed in the top right corner of the stage area.
- Right Panel:** Contains the "Sprite" section, which shows the "Basketball" sprite with a size of 100 and a direction of 90. Below this is the "Backdrops" section, which shows a list of backdrops with a count of 2.

# 6 - Collision with floor - first try!

The image shows the Scratch project editor for a game titled "Fall Ball INCREMENTAL". The interface is divided into several sections:

- Top Bar:** Includes the Scratch logo, a menu (Settings, File, Edit, Share, See Project Page, Tutorials, Save Now), and a user profile (ssardina).
- Left Panel:** Contains tabs for Code, Costumes, and Sounds. Under the Code tab, there are categories for Variables, Motion, Looks, Sound, Events, Control, Sensing, and Operators. The "Variables" category is selected, showing a list of variables: "ball\_y" (checked) and "my variable" (unchecked). Below this is a "My Blocks" section with a "Make a Block" button.
- Code Area:** Displays a script for the "ball\_y" variable. The script starts with a "when green flag clicked" event, followed by "set ball\_y to 150", "go to x: 0 y: ball\_y", and a "forever" loop. Inside the loop, it "change ball\_y by -5", checks if "ball\_y < -80", and if true, "set ball\_y to -80". Finally, it "go to x: 0 y: ball\_y".
- Stage:** Shows a preview of the game. It features a red brick wall background and a grey floor. A basketball sprite is positioned at the top right. The "ball\_y" variable is displayed as a text box with the value 30.
- Bottom Panel:** Contains the "Sprites" section with a "Basketball" sprite selected. It shows the sprite's position (x: 0, y: 150), size (100), and direction (150). There is also a "Backdrops" section showing a "Basketball" backdrop.



# 7 - Collision with floor - works!

The image shows the Scratch project editor for a game titled "Fall Ball INCREMENTAL". The interface includes a top menu bar with options like Settings, File, Edit, and a main workspace divided into a code area on the left and a stage area on the right.

**Code Area:**

- Variables:** A variable named "ball\_y" is defined and set to 150.
- Control:** A "when green flag clicked" event triggers a "go to x: 0 y: ball\_y" block.
- Forever Loop:**
  - A "change ball\_y by -5" block decreases the ball's y-coordinate.
  - An "if-then" block checks for a collision:  $\text{ball\_y} = -80$  or  $\text{ball\_y} < -80$ . If true, it sets "ball\_y" to -80.
  - A "go to x: 0 y: ball\_y" block resets the ball's position after the collision.

**Stage Area:**

- The stage background is a red brick wall.
- A "Basketball" sprite is positioned on a grey floor at the bottom of the stage.
- The "Sprite" panel shows the "Basketball" sprite with a size of 100 and a direction of 150.
- The "Stage" panel shows the "Basketball" sprite with a size of 100 and a direction of 150.



# 8 Abstract velocity: ball\_vel

The image shows a Scratch project titled "07 Fall Ball INCREMENT". The interface includes a top menu bar with options like Settings, File, Edit, Share, and See Project Page. The left sidebar contains categories for Variables, Motion, Looks, Sound, Events, Control, Sensing, Operators, and My Blocks. The main workspace displays a script for a basketball sprite. The script begins with a "when green flag clicked" event, followed by setting "ball\_y" to 150 and "ball\_vel" to -5. It then moves the sprite to x=0 and y=ball\_y. A "forever" loop contains a "change ball\_y by ball\_vel" block, a "if ball\_y < -80 or ball\_y > 80 then" conditional block that sets ball\_y to 80, and a "go to x: 0 y: ball\_y" block. The right side of the interface shows a preview of the stage with a brick wall background and a basketball sprite. The sprite's properties are set to "Basketball", x=0, y=150, size=100, and direction=150. The stage backdrop is labeled "Basketball" and has a count of 2.

Scratch Project: 07 Fall Ball INCREMENT

Variables:

- ball\_y
- ball\_vel
- my variable

Code:

```
when green flag clicked
  set ball_y to 150
  set ball_vel to -5
  go to x: 0 y: ball_y
  forever loop
    change ball_y by ball_vel
    if ball_y < -80 or ball_y > 80 then
      set ball_y to 80
    go to x: 0 y: ball_y
```

Sprite: Basketball

Stage: Basketball

# 9 - Acceleration: change velocity

The image displays a Scratch project titled "09 Fall Ball INCREMENT". The project features a basketball sprite falling against a red brick wall background. The code is written in the Scratch block editor, showing the following logic:

- When green flag clicked:**
  - Set `ball_y` to 150.
  - Set `ball_vel` to -5.
  - Go to x: 0 y: ball\_y.
  - Enter a **repeat** loop (duration not specified):
    - if** `ball_y < -80` **or** `ball_y < -80` **then**:
      - Set `ball_y` to -80.
      - Set `ball_vel` to 0.
    - else**:
      - Change `ball_vel` by -1.
  - Go to x: 0 y: ball\_y.

The right side of the interface shows the stage with a basketball sprite at the top. The sprite's properties are set to `ball_y: 80` and `ball_vel: 0`. The stage background is a red brick wall, and the floor is a grey path.

# 10 - Bounce back (perfect bounce)

The image shows the Scratch development environment for a project titled "09 Fall Ball INCREMENT". The interface includes a top menu bar with options like Settings, File, Edit, Share, See Project Page, Tutorials, Save Now, and a user profile. Below the menu is a toolbar with tabs for Code, Costumes, and Sounds. The left sidebar contains a "Variables" section with a "Make a Variable" button and a list of variables: "ball\_vel" (checked), "ball\_y" (checked), and "my variable" (unchecked). The main workspace displays a script for a basketball sprite. The script starts with a "when green flag clicked" event, followed by "set ball\_y to 150", "set ball\_vel to -5", and "go to x: 0 y: ball\_y". A "repeat" loop is used to simulate the bounce. Inside the loop, the "ball\_y" variable is changed by "ball\_vel", and a "if" condition checks if "ball\_x" is less than or equal to -80. If true, the "ball\_x" is set to -80, "ball\_y" is set to -80, and "ball\_vel" is set to "ball\_vel \* -1". After the loop, "ball\_y" is changed by -1, and the sprite is moved to "x: 0 y: ball\_y". The right sidebar shows the "Sprite" panel with a "Basketball" sprite, and the "Stage" panel with a brick wall background. The "Basketball" sprite is currently at x: 0, y: 150, with a size of 100 and a direction of -165.

Scratch Project: 09 Fall Ball INCREMENT

Variables:

- Make a Variable
- ball\_vel
- ball\_y
- my variable

Code:

```
when green flag clicked
  set ball_y to 150
  set ball_vel to -5
  go to x: 0 y: ball_y
  repeat (1)
    change ball_y by ball_vel
    if ball_x <= -80 then
      set ball_x to -80
      set ball_y to -80
      set ball_vel to ball_vel * -1
    end
  end
  change ball_y by -1
  go to x: 0 y: ball_y
```

Sprite: Basketball

Stage: Basketball

# 11 - Sound and roll...

Scratch interface showing a project titled "09.Fall Ball INCREMENT". The project is in the "Code" tab, displaying a script for a falling ball game.

**Script:**

- when green flag clicked
  - set ball\_y to 150
  - set ball\_vel to -5
  - go to x: 0 y: ball\_y
  - forever loop:
    - change ball\_y by ball\_vel
    - if ball\_y = -80 or ball\_y < -80 then:
      - set ball\_y to -80
      - set ball\_vel to ball\_vel \* -1
      - start sound basketball bounce
    - else:
      - change ball\_vel by -1
      - turn 15 degrees
    - go to x: 0 y: ball\_y

**Stage:**

- Background: Red brick wall.
- Sprite: Basketball.
- Coordinates: x: 0, y: 107.
- Size: 100.
- Direction: 0.

**Variables:**

- ball\_y: 107
- ball\_vel: -12



# 12 - Loss of energy at bounce

The image shows the Scratch project editor for a project titled "Go Fall Ball INCREMENT". The interface includes a top menu bar with options like Settings, File, Edit, Share, See Project Page, Tutorials, Save Now, and a user profile. The left sidebar contains a "Code" tab and a "Costumes" tab. The main workspace displays a script for a falling ball simulation.

**Code Blocks:**

- when green flag clicked** (Event)
- set ball\_y to 150** (Motion)
- set ball\_vel to -5** (Motion)
- go to x: 0 y: ball\_y** (Motion)
- repeat** (Control) - Loop containing:
  - change ball\_y by ball\_vel** (Motion)
  - if ball\_y < -80 or ball\_y < -80 then** (Control) - Green flag condition
    - set ball\_y to -80** (Motion)
    - set ball\_vel to ball\_vel \* 0.8** (Motion) - This block implements the loss of energy at the bounce.
    - start sound: basketball bounce** (Sound)
  - else** (Control) - Loop body
    - change ball\_vel by -1** (Motion)
    - turn 15 degrees** (Motion)
    - go to x: 0 y: ball\_y** (Motion)

**Stage View:**

- The stage background is a brick wall.
- A basketball sprite is positioned at the top center of the stage.
- The sprite's properties are: **Basketball**, **x: 0**, **y: 150**, **Size: 100**, **Direction: -150**.
- The stage has a **Backdrop: Basketball** and a **Background: 2**.

# 13 - Stabilise at floor

The image shows a Scratch project titled "09 Fall Ball INCREMENT...". The code is written in the "Code" tab and implements a basketball falling and stabilizing on a floor. The logic is as follows:

- When clicked:**
  - Set `ball_y` to 150.
  - Set `ball_vel` to -5.
  - Go to x: 0, y: ball\_y.
- Forever loop:**
  - Change `ball_y` by `ball_vel`.
  - If `ball_y < -80` or `ball_y > 80`, then:
    - Set `ball_y` to -80.
    - Set `ball_vel` to `ball_vel * 0.8`.
    - If `abs(ball_vel) < 4`, then:
      - Set `ball_vel` to 0.
  - Else:
    - Start sound: basketball bounce.
  - Else:
    - Change `ball_vel` by -1.
    - Turn 15 degrees.
    - Go to x: 0, y: ball\_y.

The right side of the interface shows a stage with a red brick wall background and a grey floor. A basketball sprite is positioned at the top of the frame. The sprite's properties are set to "Basketball", size 100, and direction 185. The stage has 2 backdrops.



# 14 - Any initial location

The image shows a Scratch project titled "09.Fall Ball INCREMENT". The project is set to "09.Fall Ball INCREMENT" and is currently in the "Code" tab. The project features a basketball sprite on a brick wall background. The code is as follows:

```
when green flag clicked
  set ball_y to 0
  set ball_vel to 0
  go to x: x position y: ball_y
  forever loop
    change ball_y by ball_vel
    if ball_y < -80 or ball_y > 80 then
      set ball_y to 80
      set ball_vel to ball_vel * 0.8
      if not (not ball_vel < 0) then
        set ball_vel to 0
      else
        start sound: basketball bounce
    end
  end
  change ball_vel by -1
  turn 15 degrees
  go to x: x position y: ball_y
```

The code is written in a Scratch script. It starts with a "when green flag clicked" event. The first block is "set ball\_y to 0". The second block is "set ball\_vel to 0". The third block is "go to x: x position y: ball\_y". This is followed by a "forever" loop. Inside the loop, the first block is "change ball\_y by ball\_vel". The second block is an "if" statement: "if ball\_y < -80 or ball\_y > 80 then". Inside the "if" statement, the first block is "set ball\_y to 80". The second block is "set ball\_vel to ball\_vel \* 0.8". The third block is an "if" statement: "if not (not ball\_vel < 0) then". Inside this "if" statement, the first block is "set ball\_vel to 0". The second block is "start sound: basketball bounce". The "if" statement is followed by an "end" block. After the "if" statement, the first block is "change ball\_vel by -1". The second block is "turn 15 degrees". The third block is "go to x: x position y: ball\_y". The "forever" loop ends with a "forever" loop block.

The project also includes a "Sprite" panel showing a "Basketball" sprite. The "Stage" panel shows a "Basketball" sprite. The "Background" panel shows a "Basketball" background.

# 14 - Abstract acceleration & floor

The image shows a Scratch project titled "09 Fall Ball INCREMENT" with a purple header bar. The interface includes a left sidebar with categories: Motion, Looks, Sound, Events, Control, Sensing, Operators, Variables, and My Blocks. The "Variables" category is selected, showing a list of variables: ball\_acc, ball\_vel, ball\_y, floor, and my\_variables. The "My Variables" section is expanded, showing the following values: ball\_acc = 0, ball\_vel = 1, ball\_y = 50, and floor = 50. The "My Blocks" section is also expanded, showing a "Make a List" button and a "Make a Block" button.

The main workspace displays a script for a falling ball. The script starts with a "when green flag clicked" event, followed by a "set ball\_y to y position" block. It then sets ball\_vel to -5 and ball\_acc to -1. A "go to x: x position, y: ball\_y" block is used to move the ball to its starting position. A "forever" loop contains the following blocks: "change ball\_y by ball\_vel", "change ball\_acc by 1", a "if ball\_y > floor" condition, and a "if ball\_y > floor" condition. Inside the "if" blocks, the ball\_y is set to floor, ball\_vel is set to 0, and ball\_acc is set to 0. The script ends with a "start sound: basketball bounce" block.

The stage preview on the right shows a red brick wall and a grey floor. A basketball sprite is positioned on the floor. The "Sprite" panel shows the "Basketball" sprite, and the "Stage" panel shows the "Basketball" stage.