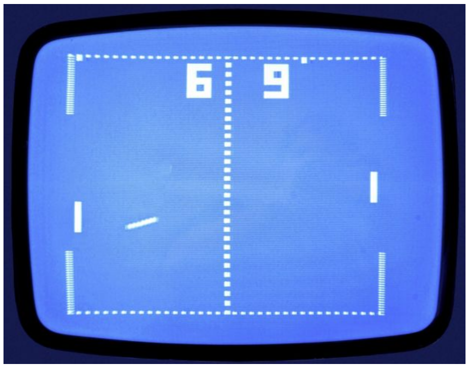
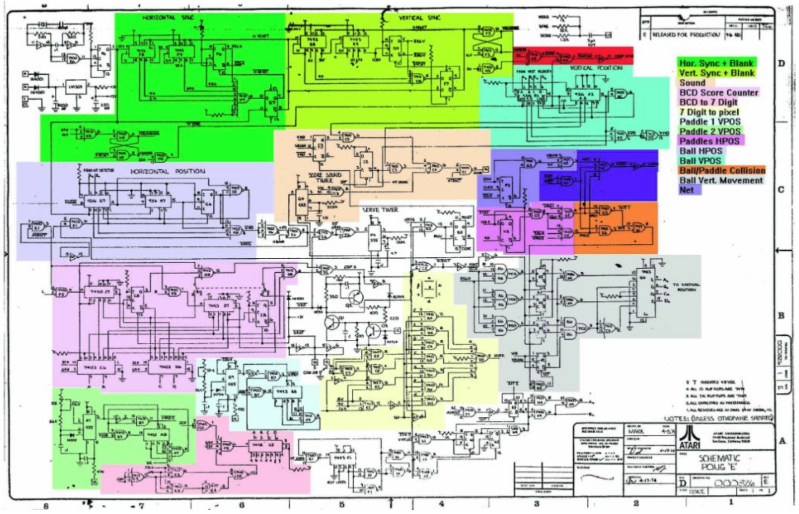
Chapter 3
Arcade Games: Force and Motion

Glen Bull and Michael Littman

Pong was the first commercially successful video game. Pong was an electronic table tennis game released by Atari and helped establish the video game industry. Due to its cultural significance, Pong is now part of the Smithsonian's permanent collections.



When the original Pong game was developed by Atari, the game was created using discrete digital logic chips. In the color-coded schematic, the logic chips responsible for the vertical and horizontal positions of the game paddles are highlighted in green and purple. The logic chips that detect a collision between the ball and a paddle are highlighted in orange.

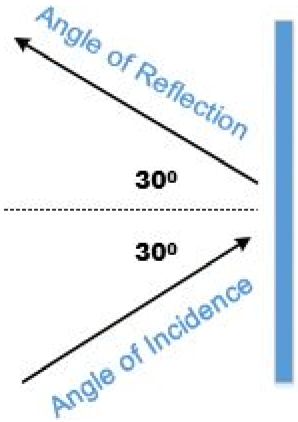


When the Atari Pong game was developed in 1972, computers were too expensive to devote a single computer to a game. Today small microcomputers like the Raspberry Pi microcomputer capable of implementing the Pong can be purchased for $25.

In this module, a Pong game will be created using *Snap!* Writing a program to recreate Pong provides opportunities to explore concepts related to force and motion. Key concepts include (1) motion is described by an object's speed and direction, (2) changes in motion are related to force and mass, and (3) friction is a force that opposes motion.

Ball Strikes the Paddle

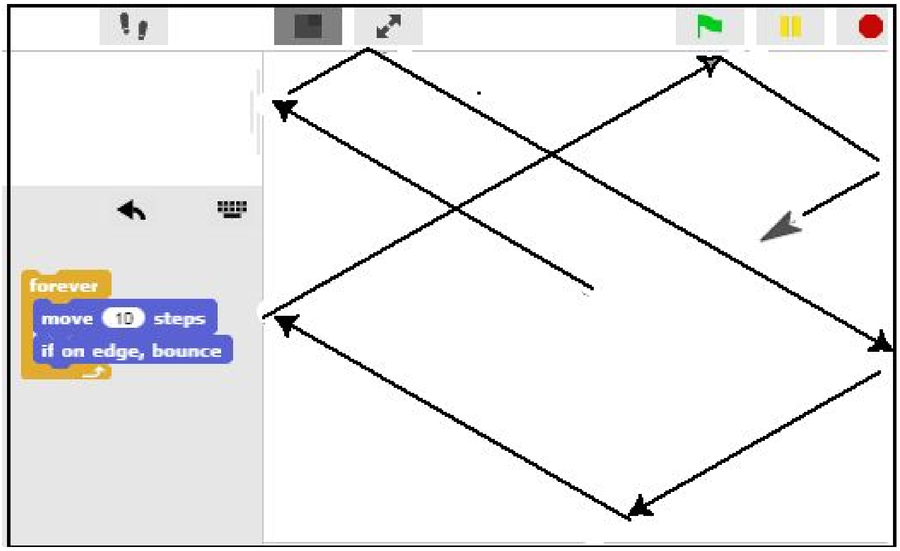
In this example, one sprite (labeled *Turtle*) will rebound when it strikes a second sprite (labeled *Paddle*). When an object such as a billiard ball strikes a surface, its angle of incidence (entry angle) when it strikes the surface is equal to its angle of reflection (exit angle) when it rebounds from the surface.



Snap! has a built-in command If on Edge, Bounce that automatically applies the correct angle of reflection when the turtle reaches the edge of the stage. This command is useful for preventing the turtle from inadvertently disappearing off the edge of the stage. The following commands will cause the turtle to travel across the stage, bouncing when it reaches an edge:

```
Forever [Move 10; If on Edge, Bounce]
```

If the pen is down, the turtle will leave a trail that makes it possible to follow its path as it travels from one edge of the screen to the next.

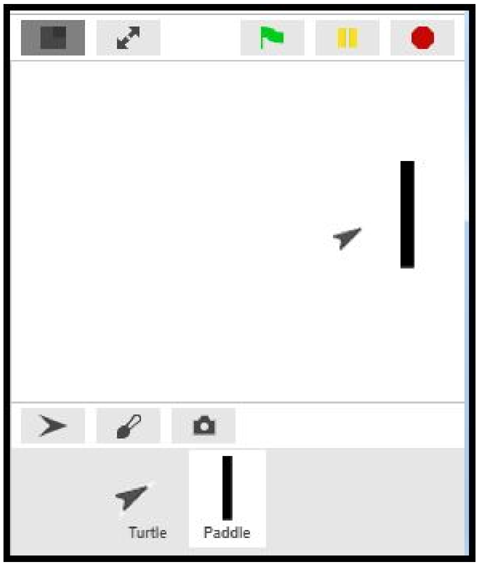


```
forever
  move (10) steps
  if on edge, bounce
```

Page Break

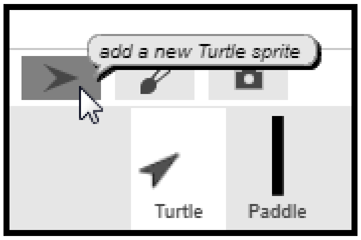
Creating the Game Paddle

In this recreation of the Pong game, the *Turtle* will travel across the screen until it strikes a second sprite (labeled *Paddle*).

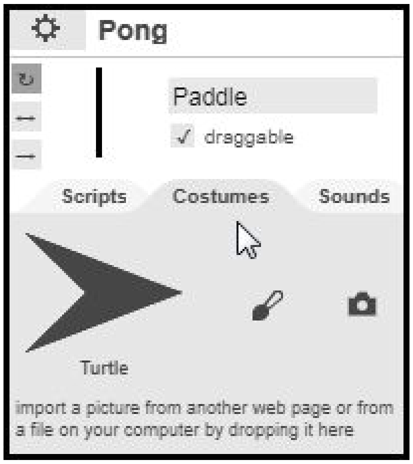


Creating the Paddle

Click on the **Sprite** icon below the stage to create a new sprite.

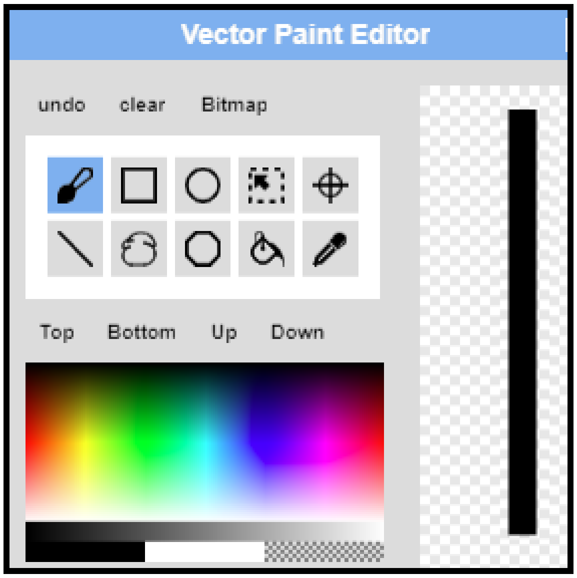


Name the newly created sprite “Paddle” and then select the *Costumes* tab to access the *Paint Editor*.



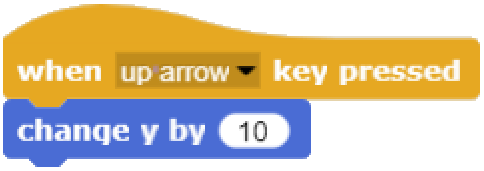
import a picture from another web page or from a file on your computer by dropping it here

Use the *Paint Editor* to create a vertical bar that will become the game paddle. The *Paint Editor* has tools to adjust the size of the game paddle as needed.

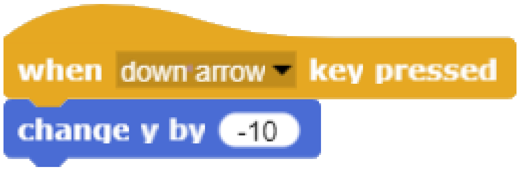


Moving the Paddle

The game paddle sprite should be placed near the back wall of the stage. The *Up* and *Down* arrow keys on the key board can be used to move the game paddle up and down. The When Up Arrow Key Pressed and the When Down Arrow Key Pressed blocks (found in the *Control Palette*) can be to determine when the up and down arrow keys have been pressed.



```
when up arrow key pressed
change y by 10
```

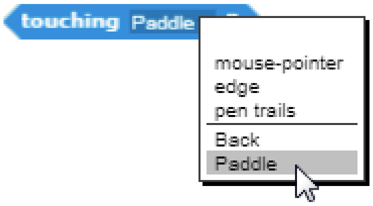


```
when down arrow key pressed
change y by -10
```

When the *up arrow* key is pressed, the Change Y by 10 block is used to move the paddle upward. When the *down arrow* key is pressed, the Change Y by -10 block is used to move the paddle downward.

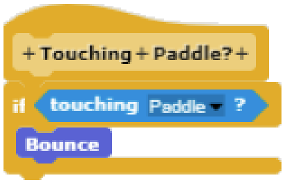
Collision Detection

When the turtle strikes the paddle, it should rebound. To cause this to occur, a means of detecting when the turtle is touching the paddle is needed. One of the commands in the *Sensing* palette can be used to detect this. A drop-down menu can be used to select other objects on the stage.



```
touching Paddle ?
```

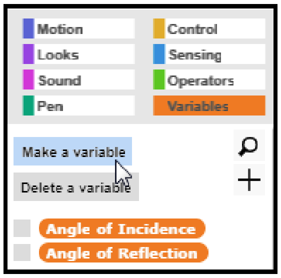
This command can be used to create a *Touching Paddle?* procedure. The *Touching Paddle?* procedure checks to see if the turtle is touching the paddle. When the turtle touches the paddle, the *Bounce* procedure is called.



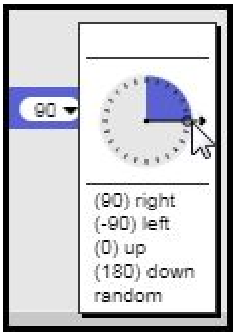
```
+Touching+Paddle?+
if touching Paddle ?
    Bounce
```

The Bounce Procedure

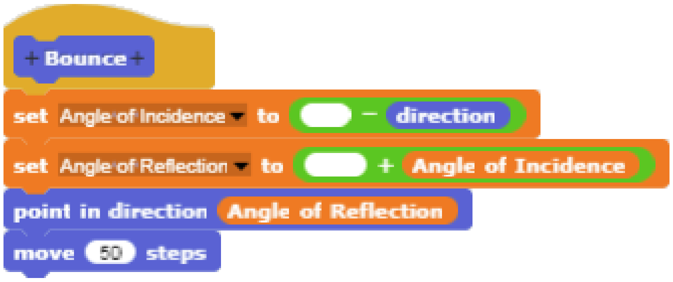
The *Bounce* procedure uses two variables: (1) Entry Angle and (2) Exit Angle. The *Entry Angle* refers to the angle of a moving object at the moment at which it strikes a surface. The *Exit Angle* refers the angle of the object at the moment at which it rebounds and exists the surface. (In geometry, the term “Angle of Incidence” is used to refer to the entry angle, and the term “Angle of Reflection” is used to refer to the exit angle.



The command *Direction* (located under the *Motion* commands palette) can be used to determine the direction in which the turtle is pointed. If the turtle is pointed straight up, the command *Direction* returns a value of 0 degrees. If the turtle is pointed straight down, the command direction returns a value of 180 degrees. If the turtle is pointed to the right, the command *Direction* returns a value of 90 degrees.



The turtle's direction can be used to determine the *Entry Angle* as it travels toward the paddle. The Entry Angle, in turn, can be used to determine the *Exit Angle*. When the *Touching Paddle?* command detects that the turtle has touched the paddle, the *Bounce* procedure uses the *Exit Angle* to point the turtle in the right direction as it rebounds.

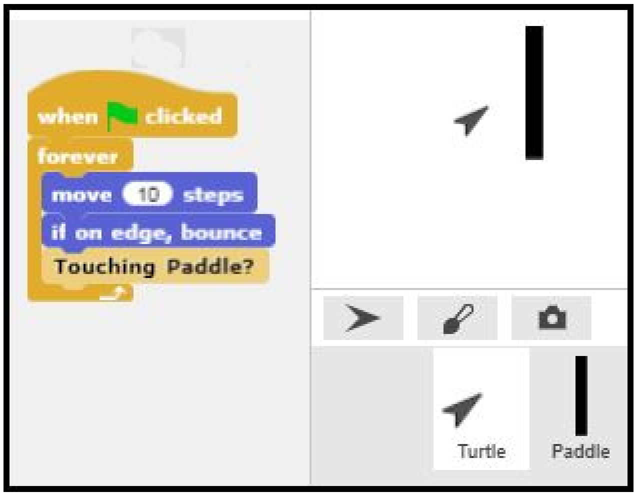


```
+ Bounce +
set Angle of Incidence to ( ) − direction
set Angle of Reflection to ( ) + Angle of Incidence
point in direction Angle of Reflection
move 50 steps
```

The *Bounce* procedure essentially performs the same calculations when the paddle is touched as the built-in *If on Edge, Bounce* command uses when the edge of the stage is reached. By running the procedure with the pen down, it is possible to determine if the angle of reflection when the turtle strikes the paddle is the same as the angle of reflection when the turtle reaches the edge of the stage and rebounds. If both angles of reflection (for the edge of the stage and the paddle) are the same, then the variable Angle of Reflection has been correctly calculated.

The Master Procedure

The master procedure (located in the turtle's script space) should look like this. The turtle should be turned to a starting position in which it is facing the paddle at an angle, as shown in the illustration.



```
when clicked
forever
  move 10 steps
  if on edge, bounce
  Touching Paddle?
```

When the *Green* go flag is clicked, the master procedure will move the turtle across the screen until it touches the paddle or the edge of the screen. When the turtle touches the paddle, the *Bounce* procedure will cause it to rebound.