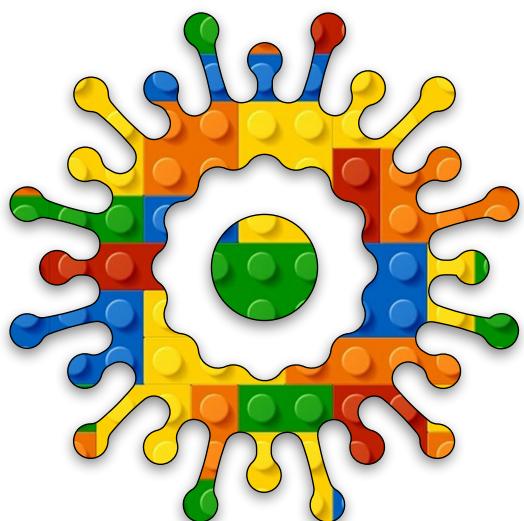


# Making Games

## Module A

### Unit #2

# pong





### Module A Unit #2 pong game

- Sketch A2.1 make the pong
- Sketch A2.2 moving the pong
- Sketch A2.3 bouncing off the edges
- Sketch A2.4 adding a paddle
- Sketch A2.5 adding paddle B
- Sketch A2.6 moving paddle B
- Sketch A2.7 moving paddle A
- Sketch A2.8 limiting the paddles
- Sketch A2.9 pong hits paddle
- Sketch A2.10 respawn the pong
- Sketch A2.11 widening the canvas
- Sketch A2.12 centre line
- Sketch A2.13 refactoring
- Sketch A2.14 the game function
- Sketch A2.15 the winner is...
- Sketch A2.16 starting and restarting the game



## Introduction to Pong

This is the ultimate classic video game. It looks so simple, and yet there is a lot you can learn about making games and some of the challenges. This can be a two-person game or single-player.



## Sketch A2.1 make the pong

This will be the pong square ball.

```
let x = 100
let y = 100

function setup()
{
    createCanvas(600, 400)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
}
```

Figure A2.1

The screenshot shows the p5.js web editor interface. At the top, there's a toolbar with icons for play/pause, stop, refresh, and a dropdown menu for "Making Games". The title bar says "sketch.js" and "Hello, TheH". The status bar indicates "Saved: about 24 hours ago" and "p5.js 1.11". The main area has two panes: the left pane contains the code editor with the following sketch.js code, and the right pane is a "Preview" canvas.

```
1 let x = 100
2 let y = 100
3
4 function setup()
5 {
6   createCanvas(600, 400)
7 }
8
9 function draw()
10 {
11   background(150)
12   noStroke()
13   square(x, y, 10)
14 }
```

The preview canvas shows a single small white square centered at coordinates (100, 100) on a gray background.



## Sketch A2.2 moving the pong

Let's make it move.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3

function setup()
{
    createCanvas(600, 400)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
}
```

Figure A1.2

The screenshot shows the p5.js web editor interface. At the top, there's a navigation bar with 'File', 'Edit', 'Sketch', 'Help', and language selection ('English'). Below the bar, there are icons for play/pause, stop, and refresh, followed by 'Auto-refresh' and 'Making Games'. On the right, it says 'Hello, TheHacker' and has 'Public' and 'p5.js 1.1.1' buttons. The main area has a title 'sketch.js' and a status 'Saved: just now'. The code editor contains the following JavaScript code:

```
1 let x = 100
2 let y = 100
3 let xvelocity = 5
4 let yvelocity = 3
5
6 function setup()
7 {
8   createCanvas(600, 400)
9 }
10
11 function draw()
12 {
13   background(150)
14   noStroke()
15   square(x, y, 10)
16   x = x + xvelocity
17   y = y + yvelocity
18 }
```

To the right of the code editor is a preview window showing a dark gray canvas with a small white square at the center. At the bottom left is a 'Console' tab and at the bottom right is a 'Clear' button.



## Sketch A2.3 bouncing off the edges

Now to bounce off the walls.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3

function setup()
{
    createCanvas(600, 400)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
}

function edges()
{
    if (x > width)
    {
        xvelocity = -xvelocity
    }
    if (y > height)
    {
```

```
yvelocity = -yvelocity  
}  
if (x < 0)  
{  
    xvelocity = -xvelocity  
}  
if (y < 0)  
{  
    yvelocity = -yvelocity  
}  
}
```

Figure A2.3

The screenshot shows the p5.js code editor interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right of the menu is a status bar showing "Hello, TheH" and "p5.js 1.11". Below the toolbar, the title bar says "sketch.js" and "Saved: 2 minutes ago". On the right side, there's a preview window titled "Preview" showing a dark gray canvas. The main area contains the following JavaScript code:

```
1 let x = 100
2 let y = 100
3 let xvelocity = 5
4 let yvelocity = 3
5
6 function setup()
7 {
8   createCanvas(600, 400)
9 }
10
11 function draw()
12 {
13   background(150)
14   noStroke()
15   square(x, y, 10)
16   x = x + xvelocity
17   y = y + yvelocity
18   edges()
19 }
20
21 function edges()
22 {
23   if (x > width)
```

At the bottom left is a "Console" button, and at the bottom right is a "Clear" button.



## Sketch A2.4 adding a paddle

Let's add a paddle.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200

function setup()
{
    createCanvas(600, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
}

function edges()
{
    if (x > width)
    {
        xvelocity = -xvelocity
    }
}
```

```
}
```

```
if (y > height)
```

```
{
```

```
    yvelocity = -yvelocity
```

```
}
```

```
if (x < 0)
```

```
{
```

```
    xvelocity = -xvelocity
```

```
}
```

```
if (y < 0)
```

```
{
```

```
    yvelocity = -yvelocity
```

```
}
```

```
}
```

```
function paddleA()
```

```
{
```

```
    rect(20, ypaddleA, 20, 50)
```

```
}
```

Figure A2.4

The screenshot shows the p5.js web editor interface. The top bar includes the p5 logo, file navigation (File, Edit, Sketch, Help, English), and a public sharing link. The code editor on the left displays a script named 'sketch.js' with the following content:

```
function setup() {
  if (x > width) {
    xvelocity = -xvelocity
  }
  if (y > height) {
    yvelocity = -yvelocity
  }
  if (x < 0) {
    xvelocity = -xvelocity
  }
  if (y < 0) {
    yvelocity = -yvelocity
  }
}
function paddleA() {
  rect(20, y paddleA, 20, 50)
}
```

The preview window on the right shows a dark gray canvas with a small white vertical rectangle at the top center, representing the ball's position.



## Sketch A2.5 adding paddle B

Let's make the other paddle.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(600, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

}

function edges()
{
    if (x > width)
```

```
{  
    xvelocity = -xvelocity  
}  
  
if (y > height)  
{  
    yvelocity = -yvelocity  
}  
  
if (x < 0)  
{  
    xvelocity = -xvelocity  
}  
  
if (y < 0)  
{  
    yvelocity = -yvelocity  
}  
}  
  
  
function paddleA()  
{  
    rect(20, ypaddleA, 20, 50)  
}  
  
  
function paddleB()  
{  
    rect(580, ypaddleB, 20, 50)  
}
```

Figure A1.5

The screenshot shows the p5.js code editor interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right of the menu is a status bar showing "Hello, TheH" and "p5.js 1.1". Below the toolbar, the code editor has tabs for "sketch.js" and "Preview". The code in "sketch.js" is as follows:

```
sketch.js
Saved: 15 seconds ago
1 if (y > height)
2 {
3     yvelocity = -yvelocity
4 }
5 if (x < 0)
6 {
7     xvelocity = -xvelocity
8 }
9 if (y < 0)
10 {
11     yvelocity = -yvelocity
12 }
13
14 function paddleA()
15 {
16     rect(20, y paddleA, 20, 50)
17 }
18
19 function paddleB()
20 {
21     rect(580, y paddleB, 20, 50)
22 }
```

The "Preview" tab shows a dark gray canvas with two white vertical rectangles representing paddles. One paddle is positioned at the left edge (x=20) and the other at the right edge (x=580). There is also a small white square in the center of the canvas.



## Sketch A2.6 moving paddle B

We will move paddle up and down with the up and down arrow keys.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(600, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

    if( keyIsDown(UP_ARROW))
```

```
{  
    ypaddleB -= 5  
}  
  
if( keyIsDown(DOWN_ARROW) )  
{  
    ypaddleB += 5  
}  
}  
  
function edges()  
{  
    if (x > width)  
    {  
        xvelocity = -xvelocity  
    }  
    if (y > height)  
    {  
        yvelocity = -yvelocity  
    }  
    if (x < 0)  
    {  
        xvelocity = -xvelocity  
    }  
    if (y < 0)  
    {  
        yvelocity = -yvelocity  
    }  
}
```

```
function paddleA()
{
    rect(20, y paddleA, 20, 50)
}

function paddleB()
{
    rect(580, y paddleB, 20, 50)
}
```

## Notes

Notice that the paddle does not stop at the bottom. We will address that later.

Figure A2.6

The screenshot shows the p5.js code editor interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right of the menu is a status bar showing "Hello, TheH" and "p5.js 1.1". Below the toolbar, the code editor has tabs for "sketch.js" and "Preview". The code in "sketch.js" is as follows:

```
40> sketch.js•
41  if (y > height)
42  {
43    yvelocity = -yvelocity
44  }
45  if (x < 0)
46  {
47    xvelocity = -xvelocity
48  }
49  if (y < 0)
50  {
51    yvelocity = -yvelocity
52  }
53 }
54
55 function paddleA()
56 {
57   rect(20, y paddleA, 20, 50)
58 }
59
60 function paddleB()
61 {
62   rect(580, y paddleB, 20, 50)
63 }
```

The "Preview" window shows a dark gray canvas with two white vertical rectangles representing paddles at the bottom edges. The left paddle is at x=20 and the right paddle is at x=580. The code uses the p5.js library to handle collision detection and reflection of a ball.



## Sketch A2.7 moving paddle A

Now for paddle A with the **q** and **a** keys for up and down. To make life easier, you can get the code for every key on the keyboard from this website: <https://www.toptal.com/developers/keycode>  
So **q** is **81** and **a** is **65**.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(600, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

    if( keyIsDown(UP_ARROW) )
    {
```

```
    ypaddleB -= 5
}
if( keyIsDown(DOWN_ARROW) )
{
    ypaddleB += 5
}
if( keyIsDown(81) )
{
    ypaddleA -= 5
}
if( keyIsDown(65) )
{
    ypaddleA += 5
}
}
```

```
function edges()
{
    if (x > width)
    {
        xvelocity = -xvelocity
    }
    if (y > height)
    {
        yvelocity = -yvelocity
    }
    if (x < 0)
    {
        xvelocity = -xvelocity
    }
    if (y < 0)
    {
        yvelocity = -yvelocity
    }
}
```

```
}

}

function paddleA()
{
    rect(20, yPaddleA, 20, 50)
}

function paddleB()
{
    rect(580, yPaddleB, 20, 50)
}
```

## Notes

Now you should be able to move each paddle independently with the **arrow keys** for paddle B. We use keyboard keys **q** and **a** for paddle A.

Figure A2.7

The screenshot shows the p5.js code editor interface. At the top, there's a toolbar with a play button, a refresh button, and a checkbox for 'Auto-refresh'. Below the toolbar, the file name is 'sketch.js' and it was saved '20 minutes ago'. The code editor displays the following JavaScript code:

```
40 if (y > height)
41 {
42     yvelocity = -yvelocity
43 }
44 if (x < 0)
45 {
46     xvelocity = -xvelocity
47 }
48 if (y < 0)
49 {
50     yvelocity = -yvelocity
51 }
52
53 function paddleA()
54 {
55     rect(20, y paddleA, 20, 50)
56 }
57
58 function paddleB()
59 {
60     rect(580, y paddleB, 20, 50)
61 }
```

To the right of the code editor is a 'Preview' window showing a dark gray canvas. In the bottom right corner of the preview window, there is a small black box containing the text 'sketch preview'. At the bottom of the editor, there's a 'Console' tab and a 'Clear' button.



## Sketch A2.8 limiting the paddles

Need to stop them going off the top and bottom.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(600, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

    if( keyIsDown(UP_ARROW) )
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
```

```
{  
    ypaddleB = 25  
}  
}  
  
if( keyIsDown(DOWN_ARROW))  
{  
    ypaddleB += 5  
    if (ypaddleB > 375)  
    {  
        ypaddleB = 375  
    }  
}  
  
if( keyIsDown(81))  
{  
    ypaddleA -= 5  
    if (ypaddleA < 25)  
    {  
        ypaddleA = 25  
    }  
}  
  
if( keyIsDown(65))  
{  
    ypaddleA += 5  
    if (ypaddleA > 375)  
    {  
        ypaddleA = 375  
    }  
}  
  
}  
  
function edges()  
{  
    if (x > width)
```

```
{  
    xvelocity = -xvelocity  
}  
  
if (y > height)  
{  
    yvelocity = -yvelocity  
}  
  
if (x < 0)  
{  
    xvelocity = -xvelocity  
}  
  
if (y < 0)  
{  
    yvelocity = -yvelocity  
}  
}  
  
  
function paddleA()  
{  
    rect(20, ypaddleA, 20, 50)  
}  
  
  
function paddleB()  
{  
    rect(580, ypaddleB, 20, 50)  
}
```

## Notes

The paddles will now stop when they reach the top and bottom.

Figure A2.8

The screenshot shows the p5.js code editor interface. At the top, there's a menu bar with File, Edit, Sketch, Help, and English. On the right side, it says "Hello, TheHa" and "p5.js 1.11.1". Below the menu, there are buttons for play/pause, stop, auto-refresh, and a link to "Making Games". The central area has a title "sketch.js" and a status bar indicating "Saved: 2 minutes ago". To the right is a preview window titled "Preview" showing a dark gray canvas with a small white square centered in the middle. The code editor itself contains the following JavaScript code:

```
1 let x = 100
2 let y = 100
3 let xvelocity = 5
4 let yvelocity = 3
5 let ypaddleA = 200
6 let ypaddleB = 200
7
8 function setup()
9 {
10   createCanvas(600, 400)
11   rectMode(CENTER)
12 }
13
14 function draw()
15 {
16   background(150)
17   noStroke()
18   square(x, y, 10)
19   x = x + xvelocity
20   y = y + yvelocity
21   edges()
22   paddleA()
23   paddleB()
```

At the bottom left is a "Console" button, and at the bottom right is a "Clear" button.



## Sketch A2.9 pong hits paddle

We need to bounce off the paddles for paddle A and B.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(600, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

    if( keyIsDown(UP_ARROW) )
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
```

```

    {
        ypaddleB = 25
    }
}

if( keyIsDown(DOWN_ARROW) )
{
    ypaddleB += 5
    if (ypaddleB > 375)
    {
        ypaddleB = 375
    }
}

if( keyIsDown(81) )
{
    ypaddleA -= 5
    if (ypaddleA < 25)
    {
        ypaddleA = 25
    }
}

if( keyIsDown(65) )
{
    ypaddleA += 5
    if (ypaddleA > 375)
    {
        ypaddleA = 375
    }
}

function edges()
{
    if (x > width)

```

```

{
    xvelocity = -xvelocity
}
if (y > height)
{
    yvelocity = -yvelocity
}
if (x < 0)
{
    xvelocity = -xvelocity
}
if (y < 0)
{
    yvelocity = -yvelocity
}
}

function paddleA()
{
    rect(20, ypaddleA, 20, 50)
    if (x == 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
    {
        xvelocity = -xvelocity
    }
}

function paddleB()
{
    rect(580, ypaddleB, 20, 50)
    if (x == 570 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
        xvelocity = -xvelocity
    }
}

```

}

## Notes

You should see the pong bounce off the paddles, but you will also note that it can bounce behind the paddles.



## Sketch A2.10 respawn the pong

Every time the pong misses the paddle, it respawns in the centre of the canvas and will randomly go left or right.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(600, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

    if( keyIsDown(UP_ARROW))
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
```

```

    {
        ypaddleB = 25
    }
}

if( keyIsDown(DOWN_ARROW) )
{
    ypaddleB += 5
    if (ypaddleB > 375)
    {
        ypaddleB = 375
    }
}

if( keyIsDown(81) )
{
    ypaddleA -= 5
    if (ypaddleA < 25)
    {
        ypaddleA = 25
    }
}

if( keyIsDown(65) )
{
    ypaddleA += 5
    if (ypaddleA > 375)
    {
        ypaddleA = 375
    }
}

function edges()
{
    if (x > width)

```

```
{  
    x = width/2  
    y = height/2  
    let direction = random(1)  
    if (direction < 0.5)  
    {  
        xvelocity = xvelocity  
    }  
    else  
    {  
        xvelocity = -xvelocity  
    }  
}  
if (y > height)  
{  
    yvelocity = -yvelocity  
}  
if (x < 0)  
{  
    x = width/2  
    y = height/2  
    let direction = random(1)  
    if (direction < 0.5)  
    {  
        xvelocity = xvelocity  
    }  
    else  
    {  
        xvelocity = -xvelocity  
    }  
}  
if (y < 0)  
{
```

```

        yvelocity = -yvelocity
    }
}

function paddleA()
{
    rect(20, ypaddleA, 20, 50)
    if (x <= 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
    {
        xvelocity = -xvelocity
    }
}

function paddleB()
{
    rect(580, ypaddleB, 20, 50)
    if (x == 570 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
        xvelocity = -xvelocity
    }
}

```

## \*\*\*\*\* Notes

This makes it very difficult to play because the pong is coming very quickly.

## Challenges

There are a number of things to make it a bit more playable.

1. The respawn pong is coming from two-thirds of the canvas, so it has longer to travel.
2. Make the canvas wider, e.g. **800**, but you will need to change some of the figures.
3. Slow the pong down a little bit.



## Sketch A2.11 widening the canvas

This option is the easiest in terms of changing code, as we simply alter the width of the canvas to **800**.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(800, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

    if( keyIsDown(UP_ARROW))
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
```

```
{  
    ypaddleB = 25  
}  
}  
  
if( keyIsDown(DOWN_ARROW))  
{  
    ypaddleB += 5  
    if (ypaddleB > 375)  
    {  
        ypaddleB = 375  
    }  
}  
  
if( keyIsDown(81))  
{  
    ypaddleA -= 5  
    if (ypaddleA < 25)  
    {  
        ypaddleA = 25  
    }  
}  
  
if( keyIsDown(65))  
{  
    ypaddleA += 5  
    if (ypaddleA > 375)  
    {  
        ypaddleA = 375  
    }  
}  
}  
  
function edges()  
{  
    if (x > width)
```

```
{  
    x = width/2  
    y = height/2  
    let direction = random(1)  
    if (direction < 0.5)  
    {  
        xvelocity = xvelocity  
    }  
    else  
    {  
        xvelocity = -xvelocity  
    }  
}  
if (y > height)  
{  
    yvelocity = -yvelocity  
}  
if (x < 0)  
{  
    x = width/2  
    y = height/2  
    let direction = random(1)  
    if (direction < 0.5)  
    {  
        xvelocity = xvelocity  
    }  
    else  
    {  
        xvelocity = -xvelocity  
    }  
}  
if (y < 0)  
{
```

```

        yvelocity = -yvelocity
    }
}

function paddleA()
{
    rect(20, ypaddleA, 20, 50)
    if (x <= 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
    {
        xvelocity = -xvelocity
    }
}

function paddleB()
{
    rect(780, ypaddleB, 20, 50)
    if (x == 770 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
        xvelocity = -xvelocity
    }
}

```

## Notes

This makes it much easier to play or at least less hard.

Figure A2.11

The screenshot shows the p5.js code editor interface. At the top, there's a navigation bar with 'File', 'Edit', 'Sketch', 'Help', and 'English'. On the right, it says 'Hello, TheHappyCoder!' and has 'Public' and 'p5.js 1.11.10' buttons. Below the nav is a toolbar with a play button, a square, and other icons. The main area has tabs for 'sketch.js' and 'Preview'. The code editor shows a script named 'sketch.js' with the following content:

```
sketch.js
94
95     yvelocity = -yvelocity
96 }
97 }
98
99 function paddleA()
100{
101   rect(20, y paddleA, 20, 50)
102   if (x <= 30 && y <= y paddleA + 25 && y >=
103   y paddleA - 25)
104   {
105     xvelocity = -xvelocity
106   }
107
108 function paddleB()
109{
110   rect(780, y paddleB, 20, 50)
111   if (x == 770 && y <= y paddleB + 25 && y >=
112   y paddleB - 25)
113   {
114     xvelocity = -xvelocity
115 }
```

The preview window shows a dark gray canvas with a small white vertical rectangle representing a paddle on the left side.



## Sketch A2.12 centre line

Adding a centre line.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200

function setup()
{
    createCanvas(800, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()

    if( keyIsDown(UP_ARROW) )
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
```

```
{  
    ypaddleB = 25  
}  
}  
  
if( keyIsDown(DOWN_ARROW))  
{  
    ypaddleB += 5  
    if (ypaddleB > 375)  
    {  
        ypaddleB = 375  
    }  
}  
  
if( keyIsDown(81))  
{  
    ypaddleA -= 5  
    if (ypaddleA < 25)  
    {  
        ypaddleA = 25  
    }  
}  
  
if( keyIsDown(65))  
{  
    ypaddleA += 5  
    if (ypaddleA > 375)  
    {  
        ypaddleA = 375  
    }  
}  
  
strokeWeight(4)  
stroke(255)  
for (let i = 0; i < height; i += 30)  
{  
    line(width/2, i, width/2, i + 10)
```

```
}

function edges()
{
    if (x > width)
    {
        x = width/2
        y = height/2
        let direction = random(1)
        if (direction < 0.5)
        {
            xvelocity = xvelocity
        }
        else
        {
            xvelocity = -xvelocity
        }
    }
    if (y > height)
    {
        yvelocity = -yvelocity
    }
    if (x < 0)
    {
        x = width/2
        y = height/2
        let direction = random(1)
        if (direction < 0.5)
        {
            xvelocity = xvelocity
        }
        else
```

```

{
    xvelocity = -xvelocity
}
}

if (y < 0)
{
    yvelocity = -yvelocity
}
}

function paddleA()
{
    rect(20, ypaddleA, 20, 50)
    if (x <= 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
    {
        xvelocity = -xvelocity
    }
}

function paddleB()
{
    rect(780, ypaddleB, 20, 50)
    if (x == 770 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
        xvelocity = -xvelocity
    }
}

```

Figure A2.12

The screenshot shows the p5.js code editor interface. The top bar includes 'File', 'Edit', 'Sketch', 'Help', and 'English'. On the right, it says 'Hello, TheHappyCoder!' and has 'Public' and 'p5.js 1.11.10' buttons. The main area has tabs for 'sketch.js' and 'Preview'. The code in 'sketch.js' is as follows:

```
100   yvelocity = -yvelocity
101 }
102 }
103 }
104
105 function paddleA()
106 {
107   rect(20, y paddleA, 20, 50)
108   if (x <= 30 && y <= y paddleA + 25 && y >=
109     y paddleA - 25)
110   {
111     xvelocity = -xvelocity
112   }
113
114 function paddleB()
115 {
116   rect(780, y paddleB, 20, 50)
117   if (x == 770 && y <= y paddleB + 25 && y >=
118     y paddleB - 25)
119   {
120     xvelocity = -xvelocity
121   }
}

```

The 'Preview' window shows a dark gray canvas with a vertical dashed white line in the center. Two small white rectangles representing paddles are visible at the top edges of the canvas.



## Sketch A2.13 refactoring

A bit of refactoring to tidy things up before more developments. Move all the key presses into their respective paddle functions and add some variables for keeping the score.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200
let playerA = 0
let playerB = 0
let score = 20
```

```
function setup()
{
    createCanvas(800, 400)
    rectMode(CENTER)
}
```

```
function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()
    strokeWeight(4)
```

```
stroke(255)
for (let i = 0; i < height; i += 30)
{
  line(width/2, i, width/2, i + 10)
}

function edges()
{
  if (x > width)
  {
    x = width/2
    y = height/2
    let direction = random(1)
    if (direction < 0.5)
    {
      xvelocity = xvelocity
    }
    else
    {
      xvelocity = -xvelocity
    }
  }
  if (y > height)
  {
    yvelocity = -yvelocity
  }
  if (x < 0)
  {
    x = width/2
    y = height/2
    let direction = random(1)
    if (direction < 0.5)
```

```

    {
        xvelocity = xvelocity
    }
    else
    {
        xvelocity = -xvelocity
    }
}

if (y < 0)
{
    yvelocity = -yvelocity
}

}

function paddleA()
{
    rect(20, ypaddleA, 20, 50)
    if (x <= 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
    {
        xvelocity = -xvelocity
    }

    if( keyIsDown(81))
    {
        ypaddleA -= 5
        if (ypaddleA < 25)
        {
            ypaddleA = 25
        }
    }

    if( keyIsDown(65))
    {
        ypaddleA += 5
    }
}

```

```
    if (ypaddleA > 375)
    {
        ypaddleA = 375
    }
}

function paddleB()
{
    rect(780, ypaddleB, 20, 50)
    if (x == 770 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
        xvelocity = -xvelocity
    }
    if( keyIsDown(UP_ARROW) )
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
        {
            ypaddleB = 25
        }
    }
    if( keyIsDown(DOWN_ARROW) )
    {
        ypaddleB += 5
        if (ypaddleB > 375)
        {
            ypaddleB = 375
        }
    }
}
```



## Sketch A2.14 the game function

Create a game function and add the scores for players. Every time that the pong goes over the edge (misses the paddle), the score is amended.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200
let playerA = 0
let playerB = 0
let score = 20

function setup()
{
    createCanvas(800, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()
    strokeWeight(4)
    stroke(255)
```

```

for (let i = 0; i < height; i += 30)
{
    line(width/2, i, width/2, i + 10)
}
game()
}

function edges()
{
    if (x > width)
    {
        playerA += 1
        x = width/2
        y = height/2
        let direction = random(1)
        if (direction < 0.5)
        {
            xvelocity = xvelocity
        }
        else
        {
            xvelocity = -xvelocity
        }
    }
    if (y > height)
    {
        yvelocity = -yvelocity
    }
    if (x < 0)
    {
        playerB += 1
        x = width/2
        y = height/2
    }
}

```

```

let direction = random(1)
if (direction < 0.5)
{
    xvelocity = xvelocity
}
else
{
    xvelocity = -xvelocity
}
}

if (y < 0)
{
    yvelocity = -yvelocity
}
}

function paddleA()
{
    rect(20, ypaddleA, 20, 50)
    if (x <= 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
    {
        xvelocity = -xvelocity
    }
    if( keyIsDown(81))
    {
        ypaddleA -= 5
        if (ypaddleA < 25)
        {
            ypaddleA = 25
        }
    }
    if( keyIsDown(65))
    {

```

```
    ypaddleA += 5
    if (ypaddleA > 375)
    {
        ypaddleA = 375
    }
}

function paddleB()
{
    rect(780, ypaddleB, 20, 50)
    if (x == 770 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
        xvelocity = -xvelocity
    }
    if( keyIsDown(UP_ARROW) )
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
        {
            ypaddleB = 25
        }
    }
    if( keyIsDown(DOWN_ARROW) )
    {
        ypaddleB += 5
        if (ypaddleB > 375)
        {
            ypaddleB = 375
        }
    }
}
```

```
function game()
{
    textAlign(CENTER, CENTER)
    textSize(25)
    fill(255)
    noStroke()
    text('PLAYER A', width/2 - 100, 20)
    text(playerA, width/2 - 50, 50)
    text('PLAYER B', width/2 + 100, 20)
    text(playerB, width/2 + 50, 50)
}
```

Figure A2.14

The screenshot shows the p5.js code editor interface. On the left is the code editor with the file 'sketch.js' containing the following code:

```
115 }  
116 }  
117 }  
118 if( keyIsDown(DOWN_ARROW))  
119 {  
120     ypaddleB += 5  
121     if (ypaddleB > 375)  
122     {  
123         ypaddleB = 375  
124     }  
125 }  
126 }  
127  
128 function game()  
129 {  
130     textAlign(CENTER, CENTER)  
131     textSize(25)  
132     fill(255)  
133     noStroke()  
134     text('PLAYER A', width/2 - 100, 20)  
135     text(playerA, width/2 - 50, 50)  
136     text('PLAYER B', width/2 + 100, 20)  
137     text(playerB, width/2 + 50, 50)  
138 }
```

On the right is the preview window showing a gray canvas with a vertical dashed line in the center. Above the line, 'PLAYER A' is at 8 and 'PLAYER B' is at 5. There are small white rectangles on either side of the dashed line.



## Sketch A2.15 the winner is...

We would like a winning screen, so when the score is **20** by either side, then the game stops. We create a boolean variable called **start** and set it to **false**. This is necessary just at the moment but will be shortly. Move **game()** function into the **draw()** function.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200
let playerA = 0
let playerB = 0
let score = 20
let start = false

function setup()
{
    createCanvas(800, 400)
    rectMode(CENTER)
}

function draw()
{
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
```

```
paddleB()
strokeWeight(4)
stroke(255)
for (let i = 0; i < height; i += 30)
{
    line(width/2, i, width/2, i + 10)
}
game()
```

```
}
```

```
function edges()
{
    if (x > width)
    {
        playerA += 1
        x = width/2
        y = height/2
        let direction = random(1)
        if (direction < 0.5)
        {
            xvelocity = xvelocity
        }
        else
        {
            xvelocity = -xvelocity
        }
    }
    if (y > height)
    {
        yvelocity = -yvelocity
    }
    if (x < 0)
    {
```

```

playerB += 1
x = width/2
y = height/2
let direction = random(1)
if (direction < 0.5)
{
    xvelocity = xvelocity
}
else
{
    xvelocity = -xvelocity
}
if (y < 0)
{
    yvelocity = -yvelocity
}
}

function paddleA()
{
rect(20, ypaddleA, 20, 50)
if (x <= 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
{
    xvelocity = -xvelocity
}
if( keyIsDown(81))
{
    ypaddleA -= 5
    if (ypaddleA < 25)
    {
        ypaddleA = 25
    }
}

```

```

}

if( keyIsDown(65))
{
    ypaddleA += 5
    if (ypaddleA > 375)
    {
        ypaddleA = 375
    }
}

function paddleB()
{
    rect(780, ypaddleB, 20, 50)
    if (x == 770 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
        xvelocity = -xvelocity
    }
    if( keyIsDown(UP_ARROW) )
    {
        ypaddleB -= 5
        if (ypaddleB < 25)
        {
            ypaddleB = 25
        }
    }
    if( keyIsDown(DOWN_ARROW) )
    {
        ypaddleB += 5
        if (ypaddleB > 375)
        {
            ypaddleB = 375
        }
    }
}

```

```
}

function game()
{
    textAlign(CENTER, CENTER)
    textSize(25)
    fill(255)
    noStroke()
    text('PLAYER A', width/2 - 100, 20)
    text(playerA, width/2 - 50, 50)
    text('PLAYER B', width/2 + 100, 20)
    text(playerB, width/2 + 50, 50)

    if (playerA == score || playerB == score && start == true)
    {
        background(100, 0, 0)
        fill(255)
        if (playerA == score)
        {
            text('PLAYER A won', width/2, height/2)
        }
        if (playerB == score)
        {
            text('PLAYER B won', width/2, height/2)
        }
    }
}
```

## Notes

You will notice that the player won displays but then the game carries on, we will fix that shortly.



## Sketch A2.16 starting and restarting the game

Clicking the mouse to start the game and to restart it after it is finished.

```
let x = 100
let y = 100
let xvelocity = 5
let yvelocity = 3
let ypaddleA = 200
let ypaddleB = 200
let playerA = 0
let playerB = 0
let score = 20
let start = false

function setup()
{
    createCanvas(800, 400)
    rectMode(CENTER)
    background(0, 100, 0)
    fill(255)
    textSize(25)
    textAlign(CENTER, CENTER)
    text('(click to start the game)', width/2, height/2)
    text('PLAYER A use Q for up and A for down', width/2, height/2 + 50)
    text('PLAYER B use the arrow keys for up and down', width/2, height/2 + 100)
    text('The first one to 20 points is the winner', width/2, height/2 - 50)
}

function mousePressed()
{
```

```

start = true
playerA = 0
playerB = 0
}

function draw()
{
  if (start == true)
  {
    background(150)
    noStroke()
    square(x, y, 10)
    x = x + xvelocity
    y = y + yvelocity
    edges()
    paddleA()
    paddleB()
    strokeWeight(4)
    stroke(255)
    for (let i = 0; i < height; i += 30)
    {
      line(width/2, i, width/2, i + 10)
    }
    game()
  }
}

function edges()
{
  if (x > width)
  {
    playerA += 1
    x = width/2
  }
}

```

```
y = height/2
let direction = random(1)
if (direction < 0.5)
{
    xvelocity = xvelocity
}
else
{
    xvelocity = -xvelocity
}
if (y > height)
{
    yvelocity = -yvelocity
}
if (x < 0)
{
    playerB += 1
    x = width/2
    y = height/2
    let direction = random(1)
    if (direction < 0.5)
    {
        xvelocity = xvelocity
    }
    else
    {
        xvelocity = -xvelocity
    }
}
if (y < 0)
{
    yvelocity = -yvelocity
```

```

    }

}

function paddleA()
{
    rect(20, ypaddleA, 20, 50)
    if (x <= 30 && y <= ypaddleA + 25 && y >= ypaddleA - 25)
    {
        xvelocity = -xvelocity
    }
    if( keyIsDown(81))
    {
        ypaddleA -= 5
        if (ypaddleA < 25)
        {
            ypaddleA = 25
        }
    }
    if( keyIsDown(65))
    {
        ypaddleA += 5
        if (ypaddleA > 375)
        {
            ypaddleA = 375
        }
    }
}

function paddleB()
{
    rect(780, ypaddleB, 20, 50)
    if (x == 770 && y <= ypaddleB + 25 && y >= ypaddleB - 25)
    {
}

```

```

    xvelocity = -xvelocity
}

if( keyIsDown(UP_ARROW) )
{
    ypaddleB -= 5
    if (ypaddleB < 25)
    {
        ypaddleB = 25
    }
}

if( keyIsDown(DOWN_ARROW) )
{
    ypaddleB += 5
    if (ypaddleB > 375)
    {
        ypaddleB = 375
    }
}

}

function game()
{
    textAlign(CENTER, CENTER)
    textSize(25)
    fill(255)
    noStroke()
    text('PLAYER A', width/2 - 100, 20)
    text(playerA, width/2 - 50, 50)
    text('PLAYER B', width/2 + 100, 20)
    text(playerB, width/2 + 50, 50)
    if (playerA == score || playerB == score && start == true)
    {
        background(100, 0, 0)
    }
}

```

```

fill(255)
if (playerA == score)
{
    text('PLAYER A won', width/2, height/2)
    text('click to play again', width/2, height/2 + 50)
}
if (playerB == score)
{
    text('PLAYER B won', width/2, height/2)
    text('click to play again', width/2, height/2 + 50)
}
if(mousePressed())
{
    start = true
}
start = false
}
}

```

## Challenges

1. Have the pong speed up after each time a paddle hits it.
2. Make the paddles get smaller as the game goes on.
3. Could you make player A be controlled by the computer, e.g. using `lerp()`?
4. Add sound each time the pong is hit.

Figure A2.16a

The screenshot shows the p5.js code editor interface. The left panel displays the sketch.js code, and the right panel shows a dark green preview area with white text instructions. The code handles game logic for two players, A and B, and includes a mouse press detection loop.

```
sketch.js
start == true
{
    background(100, 0, 0)
    fill(255)
    if (playerA == score)
    {
        text('PLAYER A won', width/2, height/2)
        text('click to play again', width/2,
height/2 + 50)
    }
    if (playerB == score)
    {
        text('PLAYER B won', width/2, height/2)
        text('click to play again', width/2,
height/2 + 50)
    }
    if(mousePressed())
    {
        start = true
    }
    start = false
}
```

The first one to 20 points is the winner  
(click to start the game)  
PLAYER A use Q for up and A for down  
PLAYER B use the arrow keys for up and down

Figure A2.16b

The screenshot shows the p5.js code editor interface. The left panel displays the sketch.js code, and the right panel shows a dark red preview area with white text indicating a win for Player B. The code remains the same as in Figure A2.16a.

```
sketch.js*
start == true
{
    background(100, 0, 0)
    fill(255)
    if (playerA == score)
    {
        text('PLAYER A won', width/2, height/2)
        text('click to play again', width/2,
height/2 + 50)
    }
    if (playerB == score)
    {
        text('PLAYER B won', width/2, height/2)
        text('click to play again', width/2,
height/2 + 50)
    }
    if(mousePressed())
    {
        start = true
    }
    start = false
}
```

PLAYER B won  
click to play again