Bonus: extra balls 3
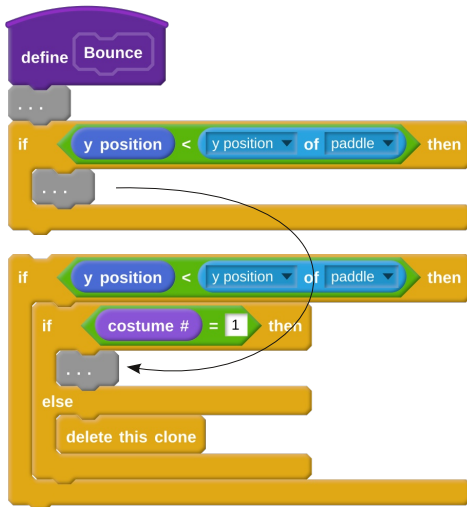
Finally, we only lose a life, when the original ball drops. The clones simply "disappear".
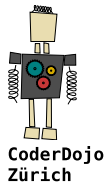
```
define Bounce
...
if  y position  <  y position ▼ of paddle ▼  then
    ...

if  y position  <  y position ▼ of paddle ▼  then
    if  costume #  = 1  then
        ...
    else
        delete this clone
```
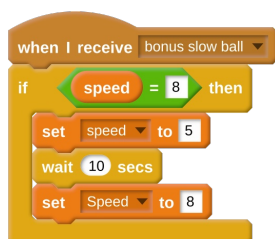
CoderDojo Zürich

Bonus: a slow ball

When the player catches a "Slow ball" bonus (the number 2) the ball gets slower.

```
if  bonus  = 2  then
broadcast  bonus slow ball ▼
```
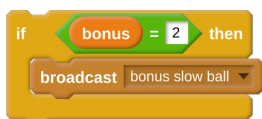
Bonus: a slow ball 2

The ball listens to the "bonus slow ball". Then, if the speed is currently the normal one, set the variable to the lower value, wait 10 seconds, before setting the speed value to its normal value.
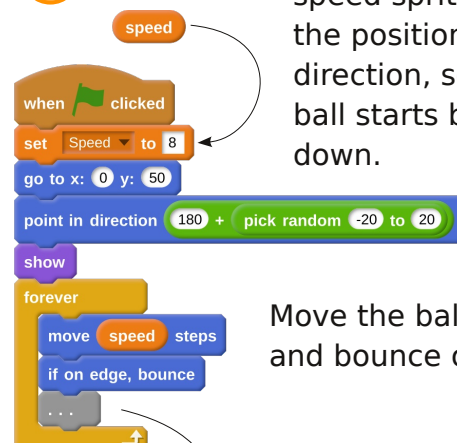
```
when I receive  bonus slow ball ▼
if  speed  = 8  then
    set  speed ▼  to 5
    wait 10 secs
    set  Speed ▼  to 8
```

Breakout

Hit all the bricks with the bouncing ball.

Move the paddle

```
when 🏳 clicked
go to x: 0 y: -150
forever
    set x to  mouse x
```

Draw a paddle or pick it from the Scratch library. Place it towards the bottom of the screen.
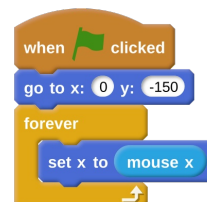
A bouncing ball

First, initialize the speed sprite variable, the position and the direction, so that the ball starts by going down.

```
speed

when 🏳 clicked
set  Speed ▼  to 8
go to x: 0 y: 50
point in direction  180 +  pick random -20 to 20
show
forever
    move  speed  steps
    if on edge, bounce
    ...
```

Move the ball at its speed and bounce on edges

See on the next page for the bouncing on the paddle...

Breakout

A ball bounces around.

With the paddle you make sure that it does not fall down to the ground.

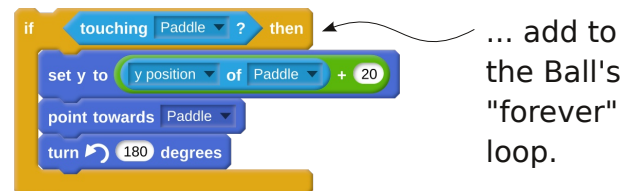The goal is to hit all the bricks with the ball.

You will need for the game:
- a ball,
- a paddle,
- a wide paddle costume,
- bricks of different colors, (one costume per color)
- pills with super powers.

Pick the sprites from the library or draw them yourself.

1

The ball und the paddle

```
if  touching Paddle ? then
    set y to  y position of Paddle  + 20
    point towards Paddle
    turn ↻ 180 degrees
```

... add to the Ball's "forever" loop.

When the ball touches the paddle it will bounce to the left or to the right, depending on which part of the paddle is touched, more or less steep, depending on how close ball is to the center of the pad:
– First, move up by 20 px so that it does not touch the paddle anymore.
– Then point to the center of the paddle.
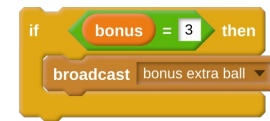– Finally invert the direction (turn by 180°)

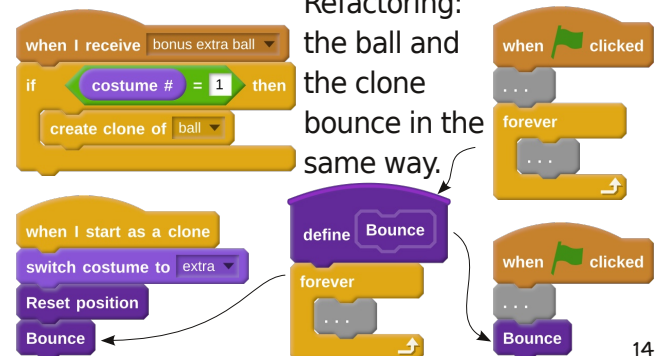Click on the green flag: the ball will bounce around and the pad follow the mouse.

Bonus: extra balls

Each time the player catches an "Extra ball" bonus (the number 3) one more ball starts bouncing around.

```
if  bonus = 3  then
    broadcast bonus extra ball
```
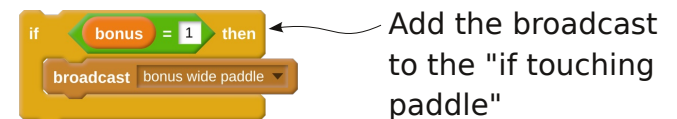
Bonus: extra balls 2

The ball listens to the "bonus extra ball". If costume is the normal one, create a clone. What does a clone do? It changes the costume to "extra", goes to the start position and starts bouncing.

```
when I receive bonus extra ball
if  costume # = 1  then
    create clone of ball
```

Refactoring: the ball and the clone bounce in the same way.

```
when I start as a clone
switch costume to extra
Reset position
Bounce
```

```
define Bounce
forever
    ...
```

```
when 🚩 clicked
...
forever
    ...
```

```
when 🚩 clicked
...
Bounce
```

14

Bonus: a wide paddle

When the player catches a "Wide Paddle" bonus (the number 1), the paddle gets wider. It will return to the normal size after 10 seconds.

```
if  bonus = 1  then
    broadcast bonus wide paddle
```

Add the broadcast to the "if touching paddle"

Bonus: a wide paddle

The paddle listens to the "Bonus Wide Paddle". It then switches to a costume with a wider paddle and waits for 10 second before switching back to the normal costume.

```
when I receive bonus wide paddle
if  costume # = 1  then
    switch costume to wide
    wait 10 secs
    switch costume to normal
```

```
when 🚩 clicked
switch costume to normal
...
```

12
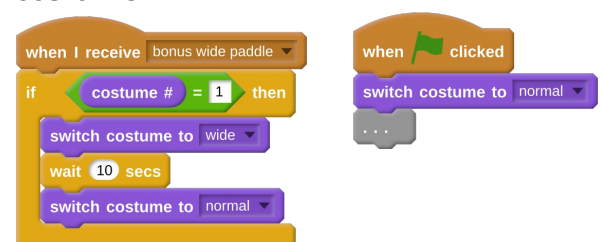
Create a "Bonus" sprite variable.
If the random value is bigger than 3 (3 chances out of 4) just do nothing and delete the clone...

bonus

```
when I start as a clone
set bonus to pick random 1 to 12
if < bonus > 3 > then
    delete this clone
```

1 to 12? We have three bonus and 1/4 chance:
3 * 4 = 12

... otherwise, move to the place where the brick was, appear there, and fall down until it gets past the paddle (or touches it).

```
switch costume to bonus
go to x: Bonus start X y: Bonus start Y
show
repeat until < y position < y position of paddle >
    change y by -6
    if < touching paddle ? > then
        ...
        delete this clone
    ↻
delete this clone
```

We need three costumes.

Broadcast from here the bonus messages...

When the ball drops below the paddle, decrease the number of lives, then the ball goes back to its start position.

balls

```
when 🏴 clicked
...
set balls to 3
```

First create the "Ball" variable (for all sprites) and initialize it to the number of lifes.

```
if < y position < y position of Paddle > then
    hide
    change balls by -1
    go to x: 0 y: 50
    point in direction 180 + pick random -20 to 20
    show
    wait 2 secs
```

In the forever loop on page 2 we add a check for the ball position being below the paddle. If it's the case, we decrease the number of lifes and – after two seconds – move back to the start position.

4

We set different colors for each row and on the second row add three gray bricks that cannot be destroyed.

```
define Set Style
switch costume to row + 1
if < row = 1 > then
    if < column = 3 or column = 6 or column = 9 > then
        switch costume to gray
```

Create 4 costumes of different colors, the forth being the gray one.

Add "Set style" just before the clone is shown...

```
When I start as a clone
...
Set Style
show
...
if < not < costume # = 4 > > then
    wait until touching ball ?
    ...
    delete this clone
```

Put an "if" around the blocks between "wait until" to "delete this clone".

In the winner, wait for 3 bricks left, not 0:

```
wait until < bricks > 4 >
wait until < bricks < 4 >
```

We keep an eye on the number of balls left and trigger a "Game over" when no we have lost all balls.

```
when 🏴 clicked
hide
wait until < balls > 1 >
wait until < balls < 1 >
show
stop all
```

First, wait until at least one brick has being cloned: otherwise it might stop as soon it starts.

We are now drawing three rows of bricks. We draw the brick zero. We keep it hidden and place somewhere in the left top corner.

```
when 🏴 clicked
hide
go to x: -210 y: 160
```

We have made some calculations and if our bricks are 30 by 15 pixels we can fit them in 13 columns and 3 rows.

6

You might have noticed, that the code for the initialization is very similar to the one for resetting the position after the ball has been missed.
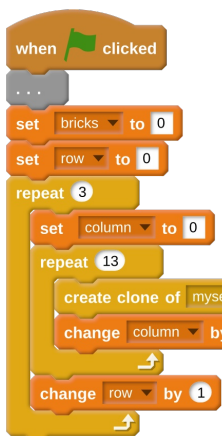Let's refactor the code and create a "Reset position" block.

```
define Reset position
go to x: 0 y: 50
point in direction 180 + pick random -20 to 20
```

We can use "Reset position" for both the "initialization" and when moving back to the start position:

```
set Speed to 8
set Balls to 3
Reset position
show
```

```
hide
change Balls by -1
wait 2 secs
Reset position
show
```

5

When the ball hits a brick, there is one chance in four to have a falling pill. If the player catches the falling pill, she/he will get a bonus.

```
bonus start X
bonus start Y
```

```
set bonus start X to x position
set bonus start Y to y position
broadcast Drop bonus
```

– Create the global variables "bonus start X" and "bonus Start Y"
– Towards the end of "When I start as a clone, just before deleting the clone...
– ... Set "bonus Start X" to the current "x position" and "bonus Start Y" to the "y position"
– Broadcast the "Drop bonus" Message.

Draw a pill. When the game starts it hides itself and when a brick broadcasts a "Drop bonus" it clones itself...

```
when 🚩 clicked
hide
```

```
when I receive Drop bonus
create clone of myself
```

10

```
when 🚩 clicked
...
set bricks to 0
set row to 0
repeat 3
  set column to 0
  repeat 13
    create clone of myself
    change column by 1
  change row by 1
```
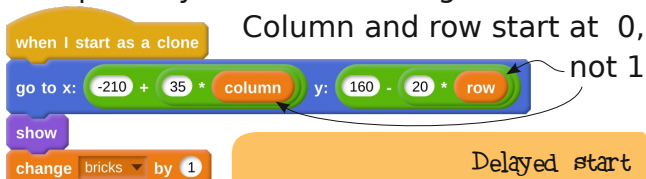
We keep track of the number of bricks, the row and the column, and we create each brick as a clone of the brick 0. Important notice: "row" and "column" must be for the sprite only. That way, each brick knows where it is.
"bricks" is a global variable.

Each column is 35 wide and the rows 20 high. Each clone position is calculated by adding to the origin the index (colum or row) multiplied by the width or height.

```
when I start as a clone
go to x: -210 + 35 * column y: 160 - 20 * row
show
change bricks by 1
```

Column and row start at 0, not 1

Add a "wait" before the ball's forever loop.

```
wait until bricks = 39
```

```
wait until touching ball ?
change bricks by -1
delete this clone
```

Add it to the "When I start as a clone "code.
When a brick is hit by the ball it disappears...

```
if touching brick ? then
  point in direction 180 - direction
```
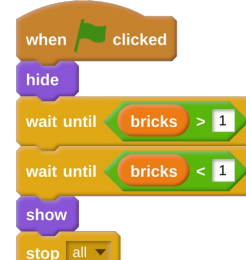
Add it to the "Forever" loop.
... And when the ball hits a brick, it inverts its direction.
(If the bricks do not disappear, wait 0.1 seconds before changing direction.)

```
Winner
when 🚩 clicked
hide
wait until bricks > 1
wait until bricks < 1
show
stop all
```

The "Winner" sprite is hidden and wait for the number of bricks being back down to 0 before showing.

First, wait for at least one bricks being cloned: otherwise it might stop as soon it starts.

8