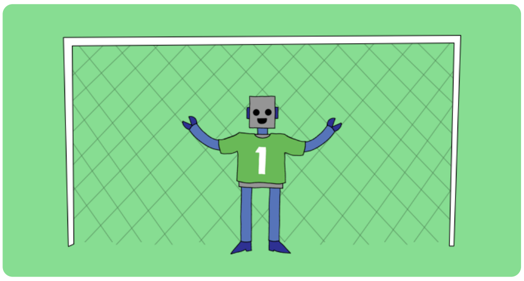
Projects

Beat the goalie

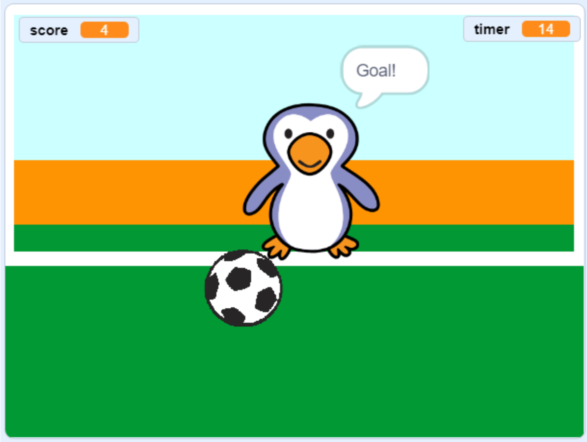
How many goals can you score in 30 seconds?



Step 1 Introduction

In this project you'll learn how to create a 2 player football game in which you have to score as many goals as you can in 30 seconds.

What you will make



What you will learn

- Recall how to use code to respond to key presses
- Use sensing blocks to detect when sprites touch each other
- Use broadcast blocks to communicate between sprites

What you will need

Hardware

- A computer capable of running Scratch 3

Software

- Scratch 3 (either **online** (http://rpf.io/scratchon) or **offline** (http://rpf.io/scratchoff))

Downloads

The starter project can be found **here** (http://rpf.io/p/en/beat-the-goalie-go).

Additional information for educators

You can find the **completed project here** (http://rpf.io/p/en/beat-the-goalie-get).

Step 2 Moving the football

Let’s code the ball to move across the bottom of the stage.

Open the ‘Beat the Goalie’ Scratch starter project.

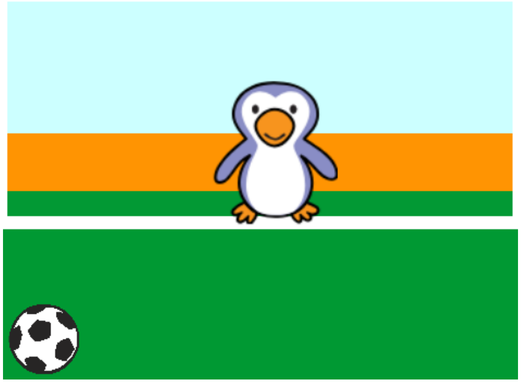
Online: open the starter project at **rpf.io/beat-the-goalie-on** (http://rpf.io/beat-the-goalie-on).

If you have a Scratch account you can make a copy by clicking **Remix**.

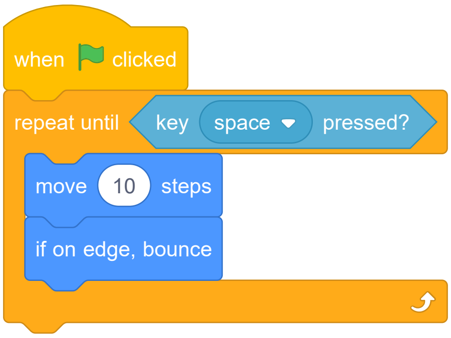
Offline: open the **starter project** (http://rpf.io/p/en/beat-the-goalie-go) in the offline editor.

If you need to download and install the Scratch offline editor, you can find it at **rpf.io/scratchoff** (http://rpf.io/scratchoff).

In the starter project, you should see a goal backdrop, a football and goalie sprite.



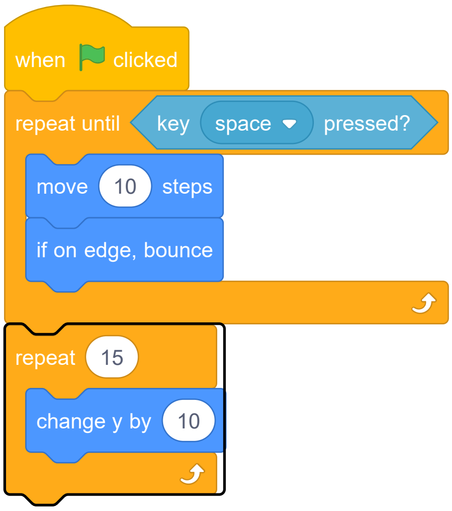
Click on your football sprite. Add this code, so that the football moves along the bottom of the screen until the space bar is pressed.



```
when flag clicked
repeat until <key (space ▾) pressed?>
    move (10) steps
    if on edge, bounce
end
```

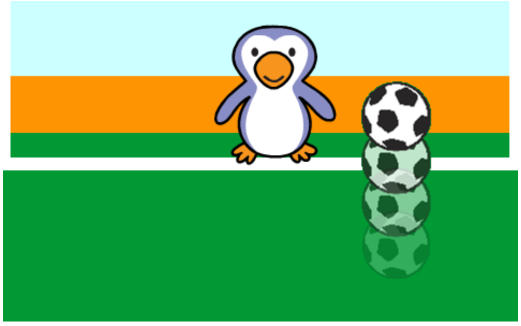
Click the green flag to test your project. Your football should bounce along the bottom of the screen until the space bar is pressed. ☑

Add this code to your football sprite, so that the football moves towards the goal after the space bar has been pressed. ☑



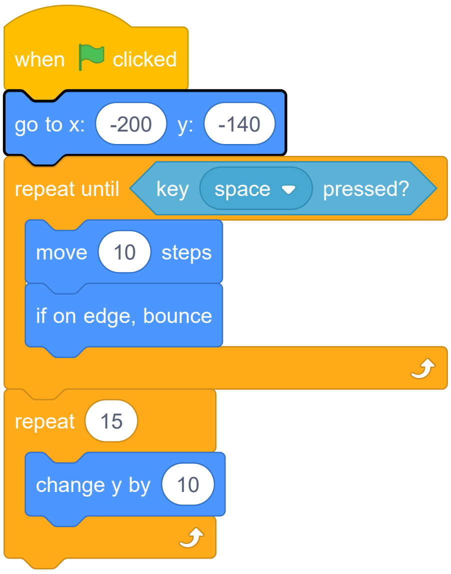
```
when 🏴 clicked
repeat until <key (space ▾) pressed?>
    move (10) steps
    if on edge, bounce
end
repeat (15)
    change y by (10)
end
```

Click the green flag to test your code. This time, press the space bar and your football should move towards the goal.



Click the green flag to test your code. What happens if you click the flag a second time? Can you fix the problem?

Your code should look like this:

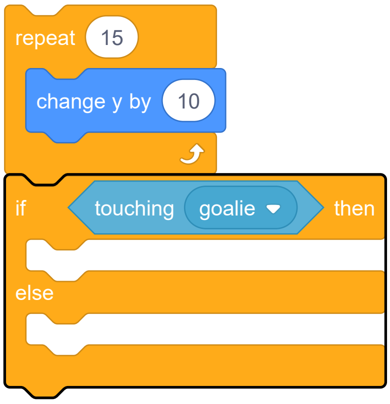


```
when 🏳 clicked
go to x: -200 y: -140
repeat until <key space ▾ pressed?>
    move 10 steps
    if on edge, bounce
end
repeat 15
    change y by 10
end
```

Step 3 Was it a goal?

Once the ball has reached the goal, there’s a decision to make. **If** the ball is touching the goalie then it has been saved, **else** it’s a goal.

Add this code to the end of your football sprite code, so that you can check whether the ball is touching the goalie. ☑

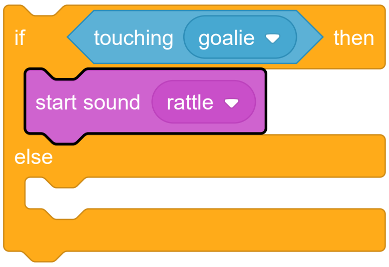


```
repeat 15
    change y by 10
if touching goalie then

else

```

Start the 'rattle' sound if the goalie has saved the ball. ☑

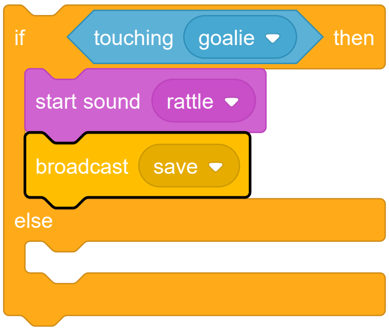


```
if touching goalie then
    start sound rattle
else

```

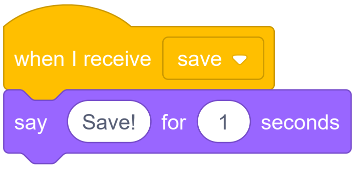
You can also broadcast a message to the goalie, so that they can tell you that the ball has been saved.

Broadcast a ‘save’ message when the ball has been saved.



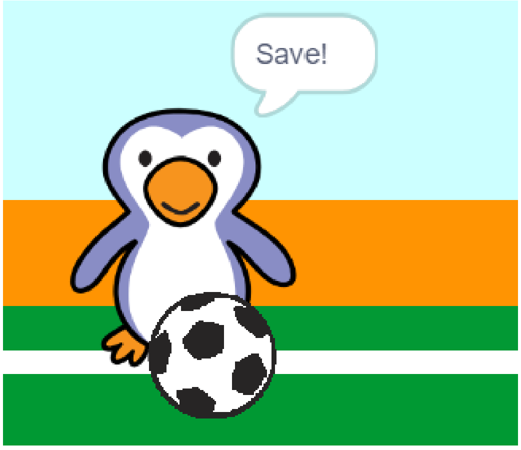
```
if <touching (goalie ▾)> then
  start sound (rattle ▾)
  broadcast (save ▾)
else
end
```

You can now code your goalie to say ‘Save!’ when they receive the message.



```
when I receive [save ▾]
say [Save!] for (1) seconds
```

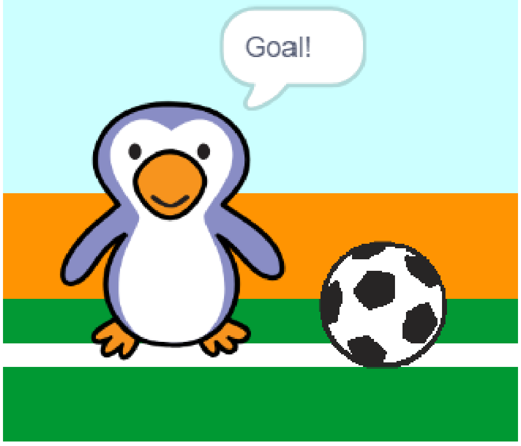
Test your code by trying to score a goal. If your goalie saves the goal they should say ‘Save!’.



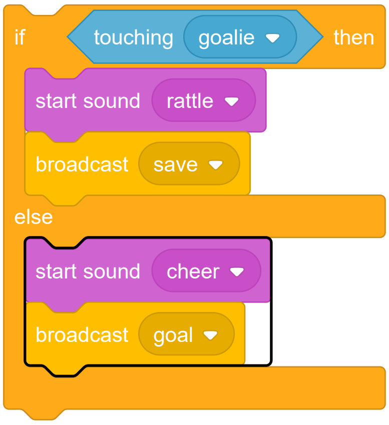
Step 4 Goal!

Can you play a sound and code your goalie to say ‘Goal!’ when a goal has been scored? ☑

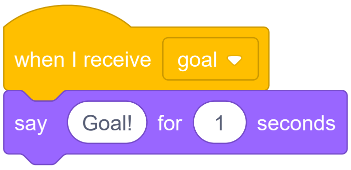
Remember that a goal has been scored if the ball is not touching the goalie.



Your code should look like this:



```
if touching goalie ▾ then
    start sound rattle ▾
    broadcast save ▾
else
    start sound cheer ▾
    broadcast goal ▾
```



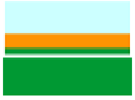
```
when I receive goal ▾
say Goal! for 1 seconds
```

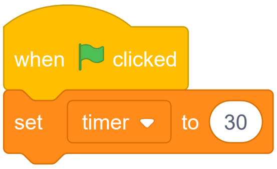
Step 5 Adding a timer

Let’s add a timer, so that the player has to score as many goals as they can in 30 seconds.

`Make a new variable` called `timer`. ☑

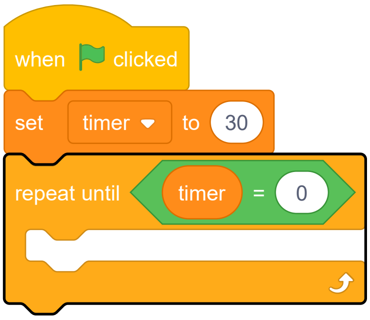
Click on your **stage**, and add this code to set the `timer` to 30 at the start of the game. ☑





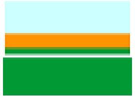
```
when ⚑ clicked
set timer ▾ to 30
```

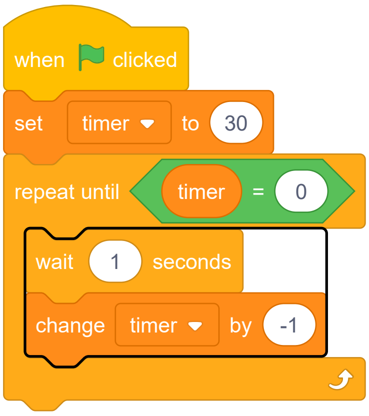
Next, you’ll need to add a `repeat until` block, so that the timer can run until it gets to 0. ☑



```
when ⚑ clicked
set timer ▾ to 30
repeat until <(timer) = 0>
    
end
```

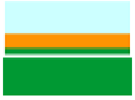
Reduce your timer by 1 every second until it reaches 0. ☑

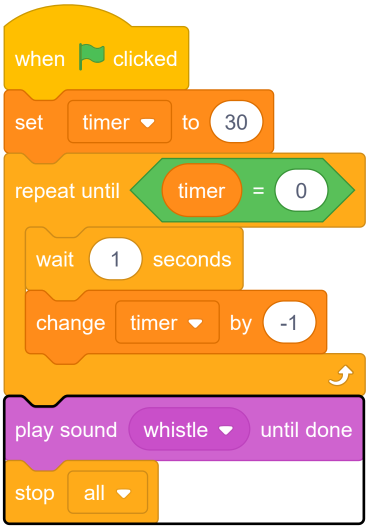




```
when ⚑ clicked
set timer ▾ to 30
repeat until <(timer) = 0>
    wait 1 seconds
    change timer ▾ by -1
end
```

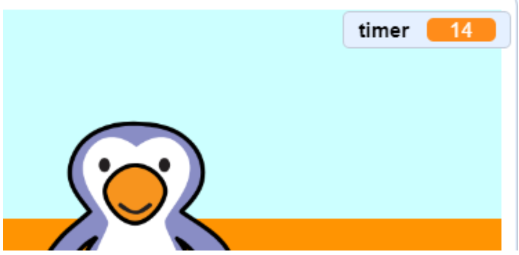
Once the timer has reached 0, you should `play the 'whistle' sound` and then stop the game. ☑





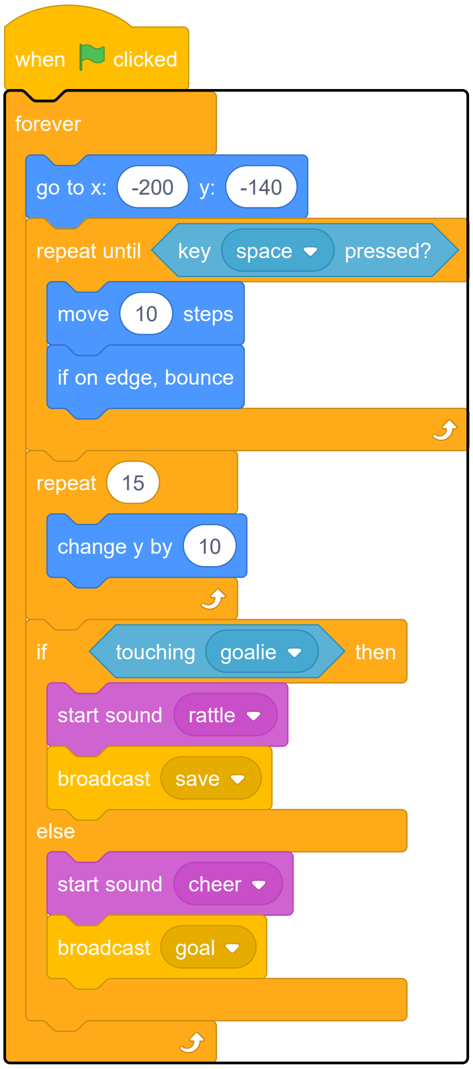
```
when ⚑ clicked
set timer ▾ to 30
repeat until <(timer) = 0>
    wait 1 seconds
    change timer ▾ by -1
end
play sound whistle ▾ until done
stop all ▾
```

Click the green flag to test your code. Your timer should start at 30, and end at 0.



You can change your timer to start at 10 if you don’t want to wait for 30 seconds!

You only have the chance to score 1 goal! To have more than 1 chance, add a `forever` block around your **football** code. You can also add a `wait` block between attempts.



```
when green flag clicked
forever
  go to x: (-200) y: (-140)
  repeat until <key (space ▼) pressed?>
    move (10) steps
    if on edge, bounce
  end
  repeat (15)
    change y by (10)
  end
  if <touching (goalie ▼)?> then
    start sound (rattle ▼)
    broadcast (save ▼)
  else
    start sound (cheer ▼)
    broadcast (goal ▼)
  end
end
```

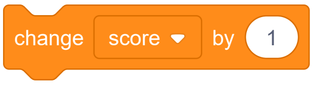
Challenge!

Challenge: adding a score

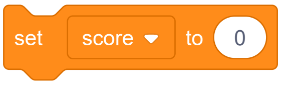
Can you add a `score` variable to your game?

- You should set the score to 0 at the start of the game;
- You should add 1 to your score every time a goal is scored.

You will need these blocks:



```
change score ▾ by 1
```

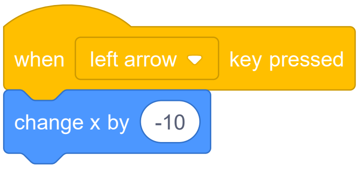


```
set score ▾ to 0
```

Step 6 Control the goalie

It’s far too easy to score a goal! Let’s allow a second player to try and save goals.

Click on your **Goalie** sprite and add this code to change the goalie’s x position when the left arrow is pressed. ☑

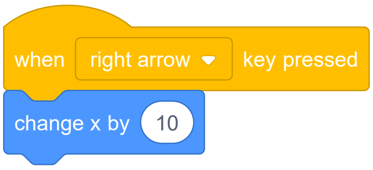


```
when [left arrow ▾] key pressed
change x by (-10)
```

Press the left arrow to test your new code. Your goalie should move to the left. ☑

Use blocks similar to the ones above to make the **Goalie** move to the right `when the right arrow key is pressed`. ☑

Your code should look like this:



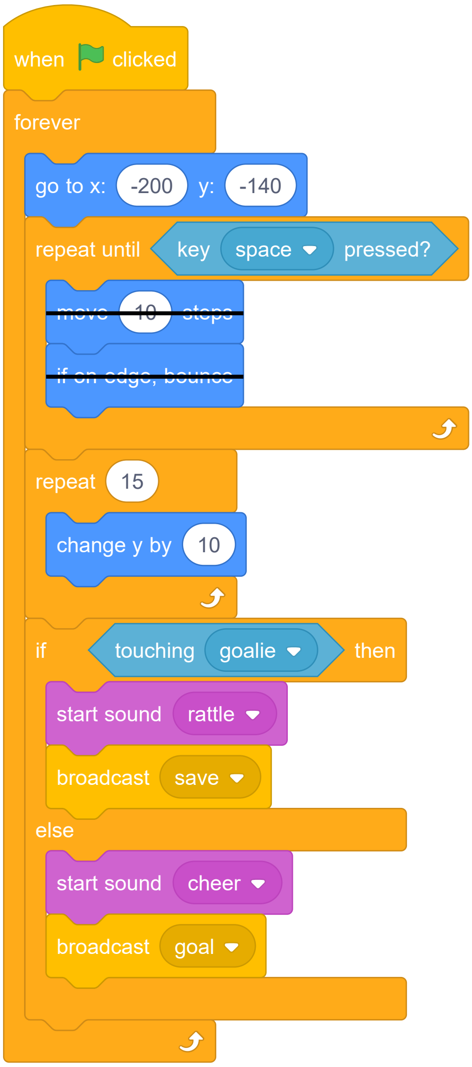
```
when [right arrow ▾] key pressed
change x by (10)
```

Challenge!

Challenge: manual control

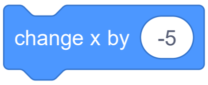
Instead of the ball moving left and right automatically, can you allow your player to control the ball with the `a` and `d` keys?

To do this you'll need to remove the code for moving the ball left and right.

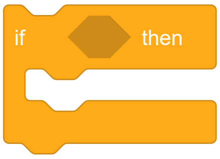


```
when 🏳 clicked
forever
    go to x: (-200) y: (-140)
    repeat until <key (space ▾) pressed?>
        ~~move (10) steps~~
        ~~if on edge, bounce~~
    end
    repeat (15)
        change y by (10)
    end
    if <touching (goalie ▾)> then
        start sound (rattle ▾)
        broadcast (save ▾)
    else
        start sound (cheer ▾)
        broadcast (goal ▾)
    end
end
```

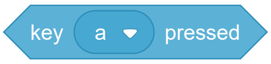
You can then add code to move the ball when the keys are pressed. Here are some code blocks to help you:



```
change x by (-5)
```



```
if <> then
end
```



```
<key (a ▾) pressed>
```

Step 7 What next?

Take a look at the **Ada's Poetry Generator** (https://projects.raspberrypi.org/en/projects/poetry-generator) Scratch project.