

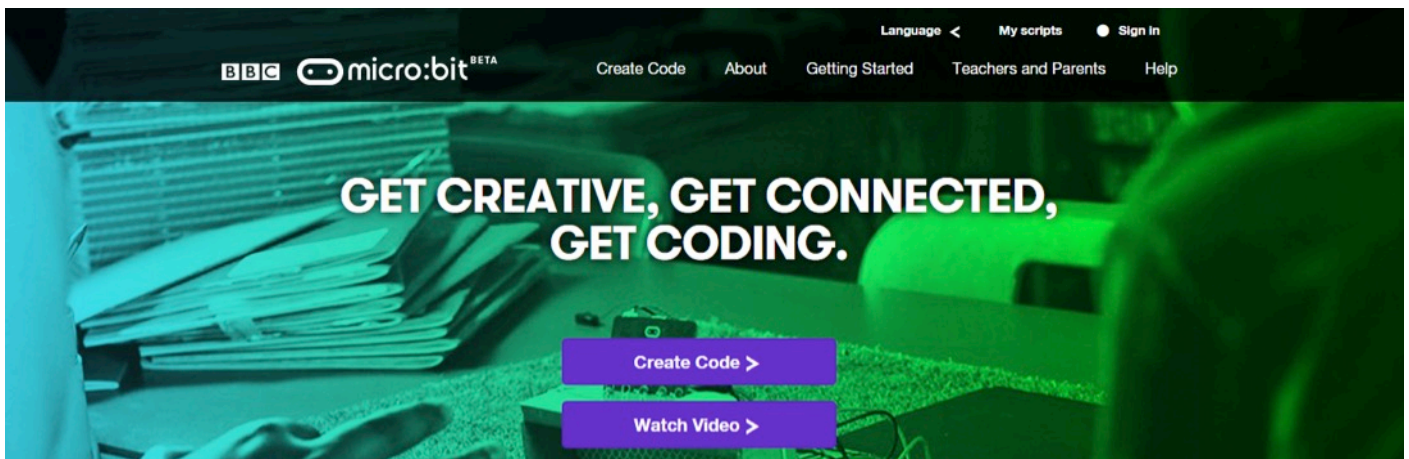


Make your very own hot potato game (basic)

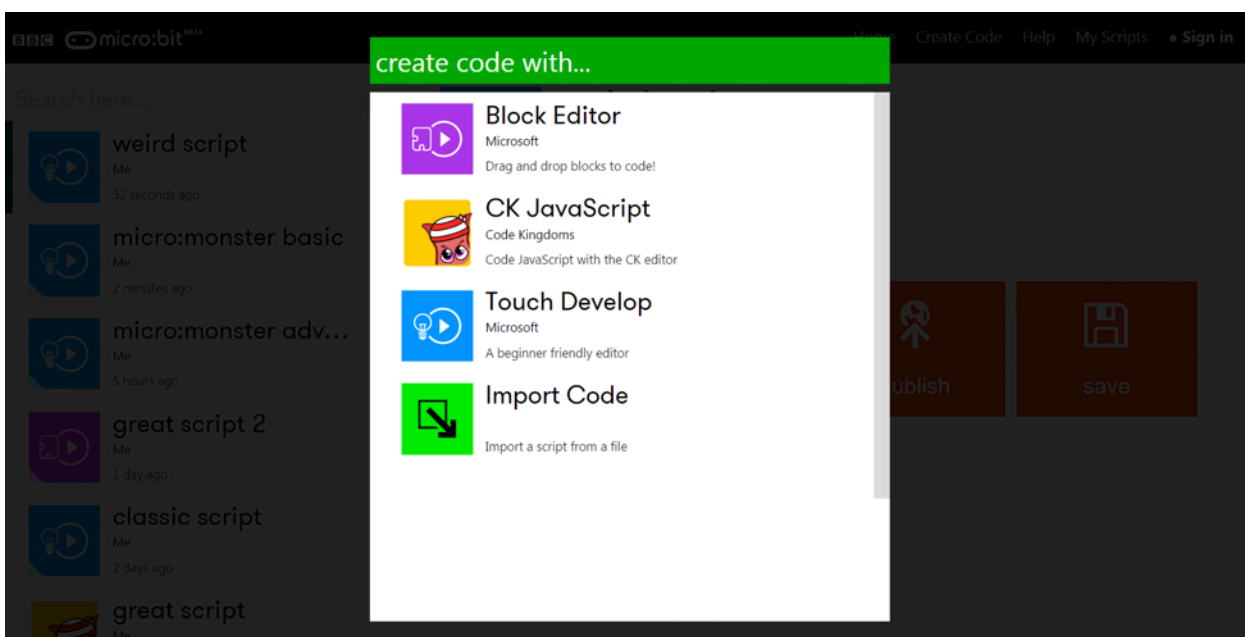
Step 1: Import the code

Download the hex file from our Live Lessons website.

Firstly, select **'My scripts'** on the top navigation on the micro:bit website (www.microbit.co.uk), and choose **'Create code'**.



Choose **'Import Code'** and upload the hex file that you've downloaded from the Live Lessons website.



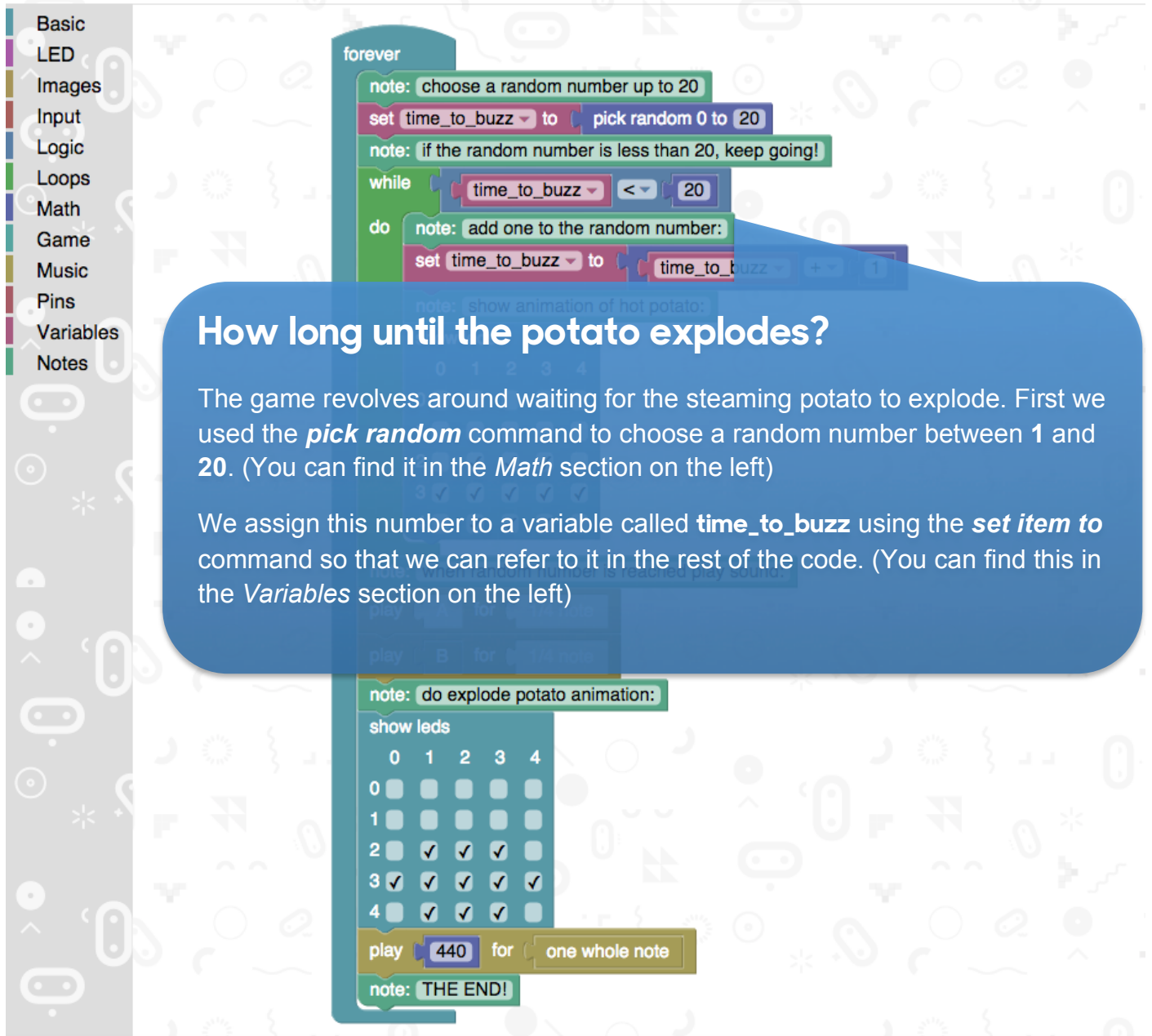
The code for your hot potato game should now appear in your code window.

Hit **'run'** to see it in action on the simulator, or plug in your micro:bit, hit **'compile'** and drag your hex file onto your micro:bit to try out your hot potato game.

Step 2: Understanding the code



hot potato basic



The screenshot shows the Micro:bit code editor interface. On the left is a sidebar with categories: Basic, LED, Images, Input, Logic, Loops, Math, Game, Music, Pins, Variables, and Notes. The main workspace contains a 'forever' loop with the following code blocks:

```
forever
  note: choose a random number up to 20
  set time_to_buzz to pick random 0 to 20
  note: if the random number is less than 20, keep going!
  while time_to_buzz < 20
  do
    note: add one to the random number:
    set time_to_buzz to time_to_buzz + 1
  note: show animation of hot potato:
  show leds
  play 440 for one whole note
  note: THE END!
```

The 'show leds' block displays a 5x5 grid of LEDs. The grid is as follows:

	0	1	2	3	4
0					
1					
2		✓	✓	✓	
3	✓	✓	✓	✓	✓
4		✓	✓	✓	

How long until the potato explodes?

The game revolves around waiting for the steaming potato to explode. First we used the *pick random* command to choose a random number between 1 and 20. (You can find it in the *Math* section on the left)

We assign this number to a variable called `time_to_buzz` using the *set item to* command so that we can refer to it in the rest of the code. (You can find this in the *Variables* section on the left)

Please note that we've removed most of the *show leds* commands so that the code fits on the page. You should see the full animation when you import the hex file you downloaded from our website.



my scripts



run



compile



convert



help

hot potato basic



- Basic
- LED
- Images
- Input
- Logic
- Loops
- Math
- Game
- Music
- Pins
- Variables
- Notes

```

forever
  note: choose a random number up to 20
  set time_to_buzz to pick random 0 to 20
  note: if the random number is less than 20, keep going!
  while time_to_buzz < 20
  do
    note: add one to the random number:
    set time_to_buzz to time_to_buzz + 1
    note: show animation of hot potato:
  show leds
  0 1 2 3 4
  0 0 0 0 0
  1 0 0 0 0
  2 0 0 0 0
  3 0 0 0 0
  4 0 0 0 0
  play 440 for one whole note
  note: THE END!

```

What happens until then?

We use a **while...do** loop (you can find it under the *Loops* section) to display the steaming potato animation.

Each time we go around the loop, we check if our variable **time_to_buzz** is less than 20.

If it is, then we continue but make sure we add one to the variable using the **set item to** command again.

We keep going around the loop, showing the animation again and again until the variable **time_to_buzz** reaches 20.



my scripts



run



compile



convert



help

hot potato basic



- Basic
- LED
- Images
- Input
- Logic
- Loops
- Math
- Game
- Music
- Pins
- Variables
- Notes

```

forever
  note: choose a random number up to 20
  set time_to_buzz to pick random
  note: if the random number is less than
  while time_to_buzz < 20
  do
    note: add one to the random number
    set time_to_buzz to time_to_buzz + 1
    note: show animation of hot potato
    show leds
      0 1 2 3 4
      0
      1
      2 ✓ ✓ ✓
      3 ✓ ✓ ✓ ✓
      4 ✓ ✓ ✓
    note: when random number is reached
    play A for 1/4 note
    play B for 1/4 note
    note: do explode potato animation:
    show leds
      0 1 2 3 4
      0
      1
      2 ✓ ✓ ✓
      3 ✓ ✓ ✓ ✓
      4 ✓ ✓ ✓
    play 440 for one whole note
    note: THE END!
  
```

Time to explode

Once our variable reaches 20, our **while... do** loop finishes. All the code beneath it tells your micro:bit what to do when the potato explodes.

First, we program it to make a noise, which we can do easily using the **play... for** command. (You can find this under *Music*.)

Next, we display our exploding animation using a series of **show leds** commands.

Finally, we play a longer note after the animation using the **play... for** command again.

Play forever..

Because we put all of our code inside a **forever** loop, our game will start again after the potato explodes.

This time it will choose a new random number between 1 and 20, so you never know how long it will be before the potato explodes.

Step 3: Modifying the code

There are lots of things you can do to adapt your hot potato game and make it your own.

You can change the longest time until the potato explodes, make your own animations for the potato or even write your own music to play at the end.

You can make it easier or harder to take care of your hot potato game, and can also change the way your hot potato game looks and acts.

Have a look at the instructions on the next page to see what you can do.

hot potato basic

my scripts run compile convert help

Basic
LED
Images
Input
Logic
Loops
Math
Game
Music
Pins
Variables
Notes

forever

note: choose a random number up to 20

set time_to_buzz to pick random 0 to 20

note: if the random number is less than 20, keep going.

while time_to_buzz <= 20

do

note: add one to the random number:

set time_to_buzz to time_to_buzz + 1

note: show animation of hot potato:

show leds	0	1	2	3	4
0					
1					
2		✓	✓	✓	
3	✓	✓	✓	✓	✓
4	✓	✓	✓	✓	✓

note: when random number is reached play sound:

play A for 1/4 note

play B for 1/4 note

note: do explode potato animation:

show leds	0	1	2	3	4
0					
1					
2		✓	✓	✓	
3	✓	✓	✓	✓	✓
4	✓	✓	✓	✓	✓

play 440 for one whole note

note: THE END!

1. Change the time to buzz
Try changing the random number from 20 to something lower and see what happens.

2. Change number in the while... do loop
If you increase the number in the while... do loop from 20, it will always take longer for your potato to explode.

3. Change the animation
You could change both of the animations to a ticking clock, countdown, or anything you like..

3. Change the sound
Why not compose your own melody to play when the time is up?

Test, play and show us what you've done

Now that you've made your very own hot potato game, click '**run**' to test it on the simulator and '**compile**' to see it working on your micro:bit.

Click '**export**' to save off your code and send it to us at live.lessons@bbc.co.uk. You could see your hot potato game featured on our **micro:bit Live Lesson** in February.

Note: If you want to be able to hear the sound your micro:bit plays when the time is up, you will need to connect a small speaker or buzzer to Pin0 and GND pins on the edge of your micro:bit.