SEAN MCMANUS

Sean McManus is a Code Club volunteer, and wrote the book *Scratch Programming in Easy Steps*. He also co-wrote *Raspberry Pi For Dummies*.
sean.co.uk
twitter.com/musicandwords

MAKE A MULTIPLE-CHOICE QUIZ IN SCRATCH

Dazzle your friends with your own quiz game, containing hundreds of questions! How many can they get right in 30 seconds?

You'll Need

> LibreOffice – Install it from the command line with: **sudo apt- get install libreoffice**

> List of capitals by size – **wki.pe/List_ of_national_ capitals_by_ population**

> Internet access

Click to answer. The answer data comes from a list on Wikipedia

The game runs for 30 seconds before it ends

L ists are used to remember lots of information, but adding items to them block by block can take a lot of time and Scratch code. In this project, you'll see how you can import (or bring in) large lists from other places, so you can easily make a quiz game with hundreds of questions. As you create this game, use your own favourite background and sprites, and arrange them with enough space for the answers to appear. Perhaps you can add your own question list? Anything works, as long as each answer only applies to one question.

>STEP-01
Gather your data
For this game, you'll need two text files: one for the questions and one for the answers. We're going to make a quiz about capital cities, so one file will contain a list of capitals, and the other will contain the countries they are in, in the same order. Start by finding the list of capital cities by population on Wikipedia. Click and drag

over the table to highlight it and then press **CTRL+C** to copy it. It's easier if you highlight from the bottom up. Be patient when the screen scrolls!
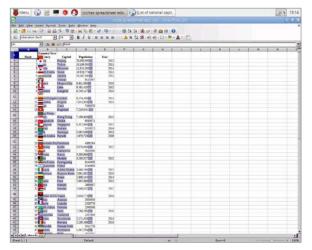
>STEP-02
Create your question files
Start LibreOffice Calc and paste in the table using **CTRL+V**. Click OK. This might take a minute or two to work. Click above your cities column to highlight it.

Above You can get a well-organised list of capital cities from this page on Wikipedia

Above Copy the capital cities table from Wikipedia into LibreOffice Calc to make it easy to extract the columns separately

Press **CTRL+C** to copy the column. Open your text editor, Leafpad, which is in the Accessories menu. Press **CTRL+V** to paste. You should now have a text file containing just capital cities, each one on a new line. If you have a heading at the top (the word 'Capital'), delete it, and remove any blank lines at the end too. Save this file as **cities.txt**. Open a new file in Leafpad and repeat the process with the countries column in LibreOffice Calc. This time, save your Leafpad file as **countries.txt**.

>STEP-03
Importing your data into Scratch
Start Scratch. Click the Variables button and make a list. Call it **cities** and make sure it's for all sprites. When the empty list appears on the Stage, right-click it and click **import** in the menu. Browse to the files you just created, and double-click your cities text file. The list on the Stage will be filled with the cities from your file. Repeat the process to make a list called **countries** and fill it with your countries file. Your list files should be the same length. Right-click the list boxes on the Stage and choose **hide**.

>STEP-04
Set up your variables
Through the Variables part of the Blocks Palette, make variables called **question number** (used to remember which question/answer pair we're asking), **score**, **shuffle choice** and **temporary storage** (used for shuffling the list of options), and **wrong answer** (used when making the list of wrong options). You also need to make a variable called **player guessed** to remember which answer the player chooses, and a list called **possible answers**. Make all these variables and the list "for all sprites".

>STEP-05
Make the main game code
The main game code uses three scripts (Listing 1). Add them all to the cat sprite. The game uses broadcasts to pass control to the various parts of the program, including on the same sprite. The 'ask a question' section picks a random question number from the list of countries and makes a list of possible answers. It includes the correct answer, and two wrong answers which must be different from the correct answer. The code then shuffles this list to put the answers in a random order, before using a broadcast to make the answer sprites appear and show their answers.

>STEP-06
Make the answer sprites
Import a new sprite to use for showing the answer. We're using Gobo. This sprite has five short scripts (Listing 2). Make the variable **answer choice**, but click the button to make it for this sprite only. If the game shows all the same answers when you run it, you probably made a mistake here! When you've finished this sprite, right-click it and duplicate it twice. In the copies, change the value of the **answer choice** variable at the top to 2 for the first one and 3 for the second one. Happy quizzing!

LISTING 1

A
```
when [flag] clicked
set score to 0
hide variable score
broadcast ask a question
reset timer
wait until <timer > 30>
broadcast game over
show variable score
stop all
```

B
```
when I receive ask a question
set question number to (pick random 1 to (length of countries))
delete all of possible answers
add item (question number) of cities to possible answers
repeat 2
    set wrong answer to question number
    repeat until < not < wrong answer = question number > >
        set wrong answer to (pick random 1 to (length of cities))
    add item (wrong answer) of cities to possible answers
repeat 5
    set shuffle choice to (pick random 1 to 3)
    set temporary storage to item (shuffle choice) of possible answers
    delete (shuffle choice) of possible answers
    insert (temporary storage) at any of possible answers
say (join [What is the capital of ] (item (question number) of countries))
broadcast show answers
```

C
```
when I receive guessed
if < item (player guessed) of possible answers = item (question number) of cities >
    say [Correct!] for 1 secs
    change score by 1
else
    say (join [The right answer is ] (item (question number) of cities)) for 2 secs
broadcast ask a question
```

LISTING 2
```
when [flag] clicked
set answer choice to 1
set size to 50 %

when I receive show answers
show
say item (answer choice) of possible answers

when Sprite2 clicked
set player guessed to answer choice
broadcast guessed

when I receive guessed
hide

when I receive game over
hide
```