

DIGITAL CREATIVES: EGYPTIAN ENIGMA

VIDEO TRANSCRIPT

PART 1: INTRODUCTION



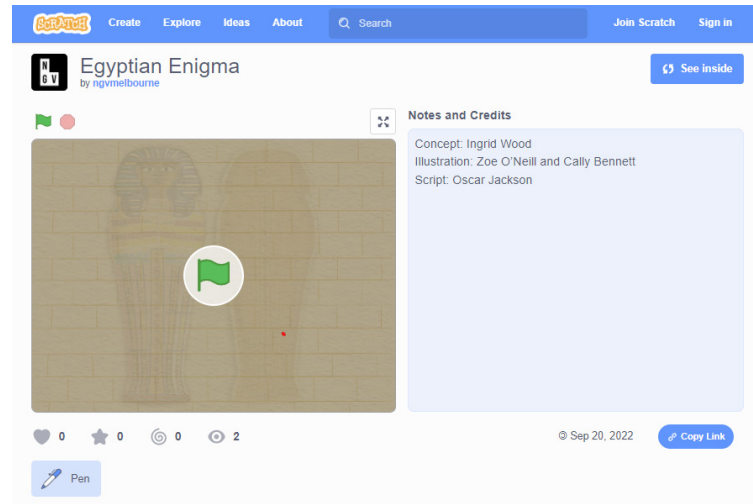
EGYPT
Inner anthropoid coffin of Tjeseb 747
BCE-589 BCE (detail)
gessoed and painted wood, linen

National Gallery of Victoria, Melbourne
Purchased from the Trustees of the
Exhibitions Building
D146-1982

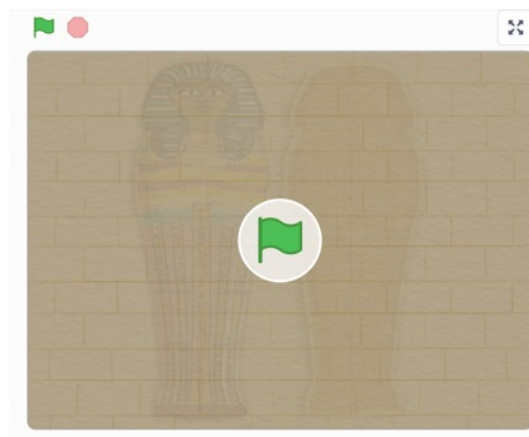
Inspired by the work undertaken by NGV conservators on the coffin of Tjeseb, we've used the block code program Scratch to create an Egyptian themed digital jigsaw puzzle.

In this video we're going to find out how the puzzle works by taking a look at the code behind it. Later, we'll add a message that displays when the puzzle is complete.

You can find the project here: <https://scratch.mit.edu/projects/707827424>



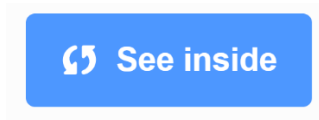
**PRESS THE GREEN FLAG
TO START (00:28)**



Press the green flag to play the game.

Our program lets us move the puzzle pieces with the cursor and snap them into place when they're in the right spot.

SEE INSIDE (00:41)



Let's take a look inside the game by pressing the **See inside** button. In Scratch, the action takes place on the **stage**. The elements on the stage are the **sprites** and the **backdrop**.

The instructions which tell the sprites and backdrop what to do are called **code blocks**. Combined together, they make the **script** which runs the program.

SPRITES AND COSTUMES (1:11)

Notice in this puzzle that even though there are lots of puzzle pieces on the stage, there are only two sprites?

Sprites are like the characters that act on the stage.

Like actors, sprites can wear different **costumes**.

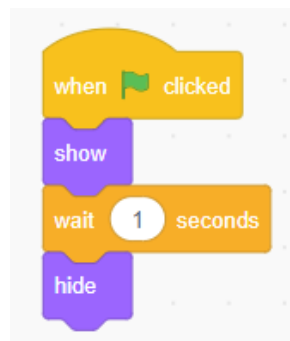
If we look at the costumes for the puzzle piece sprite, we can see that each puzzle piece is actually a costume of the original puzzle piece sprite, with its own number and position on the costume artboard, which is the same size as the stage. There are sixteen costumes for the sixteen pieces of the puzzle.

The second sprite is the puzzle area. This defines the area where our pieces are spread on the stage.

CODE ON THE SPRITES (1:57)

Let's take a look at the code on the sprites.

Puzzle area sprite:

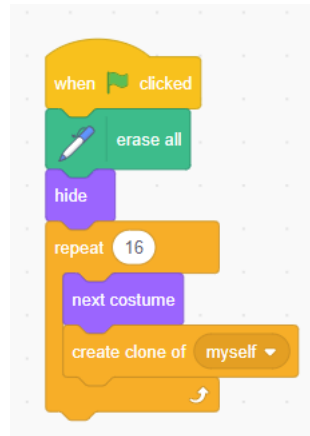


The code on the puzzle area sprite tells the sprite to show for one second when the flag is clicked and then hide.

Puzzle sprite:

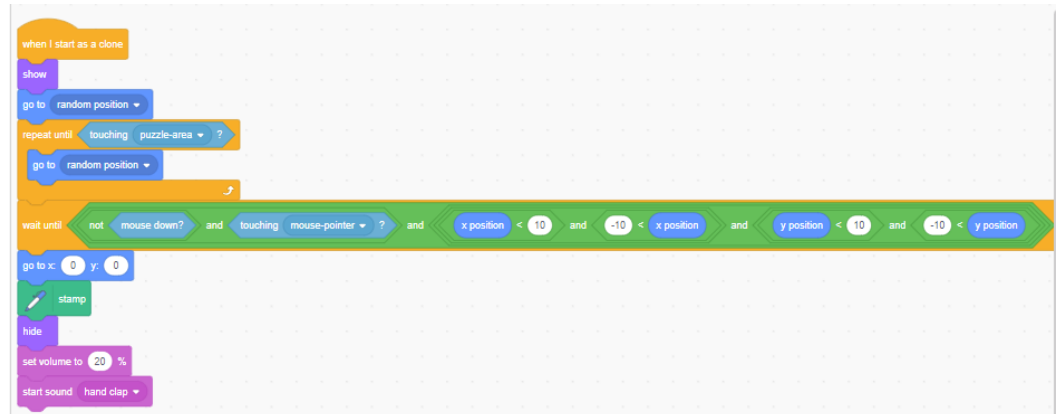
Most of the code is on the puzzle sprite.

Pen tool:



This first block of code tells the game to start when the flag is clicked and introduces the **pen tool** block. *The pen tool lets a sprite draw a trail, erase or stamp copies of itself. The pen tool erase all command, erases any pieces on the stage. The jigsaw pieces are hidden. The next instructions tell the sprite to clone itself 16 times, each time with the next costume, until all of the pieces are on the stage.

MOVING THE PUZZLE PIECES (2:49)



The second big block of code on the puzzle sprite gives instructions for moving and placing the puzzle pieces.

First it tells the program to make each cloned puzzle piece appear in a random position until it is touching the puzzle area sprite, which is really just a shape to 'contain' the pieces on the stage.

PLACING THE PIECES (3:11)

The pieces are dragged around the stage with the mouse or cursor.

The long orange control block tells the program to wait until the mouse/cursor is not down and the piece is touching the mouse pointer and the piece is within plus or minus ten pixels of its 'correct' position at x=0, y=0. If all of those things are true, to go to coordinates (0,0) stamp a copy of itself, hide the original moving piece and make a handclap sound to make it sound like it is snapping in place.

THE CORRECT POSITION (3:53)

How does the program know when each piece is in the 'correct' position?

That's a really good question! To answer it, let's look back into our sprite costumes. If we click through the costumes, we can see that each one has its own piece of the puzzle set in position with the (0,0) coordinate point in the middle (see the crosshairs).

The dimensions of the costume artboard match the stage dimensions.

When the costume artboard lines up with the stage, its position is (0,0), but the actual puzzle piece will appear in its correct position in the finished puzzle.

PART 2: WELCOME BACK

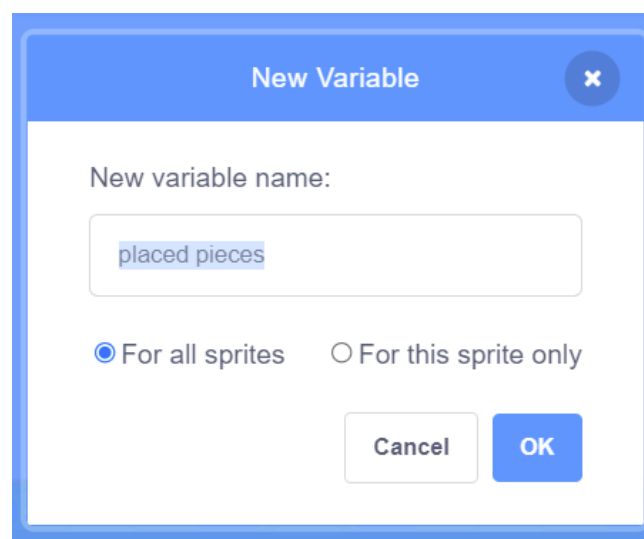
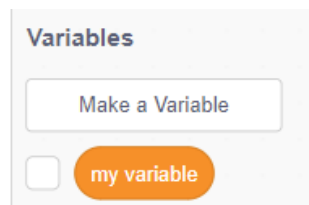
Welcome back. In our first video we unpacked the script behind our digital jigsaw to see how to generate all of our puzzle pieces, spread them randomly about the stage and how to place them using the pen stamp tool.

ADDING A MESSAGE AND SOUND (00:17)

In this video we are going to add a message and sound to congratulate the player when they complete the puzzle.

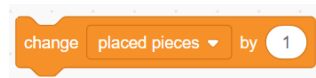
CREATE A VARIABLE TO COUNT PLACED PIECES (00:27)

We know the puzzle is complete when all 16 pieces of the puzzle have been placed. We can create a variable to count the pieces as they are placed. Click on **Make a variable**. Let's call our variable 'placed pieces'. Make sure you check **For all sprites**.

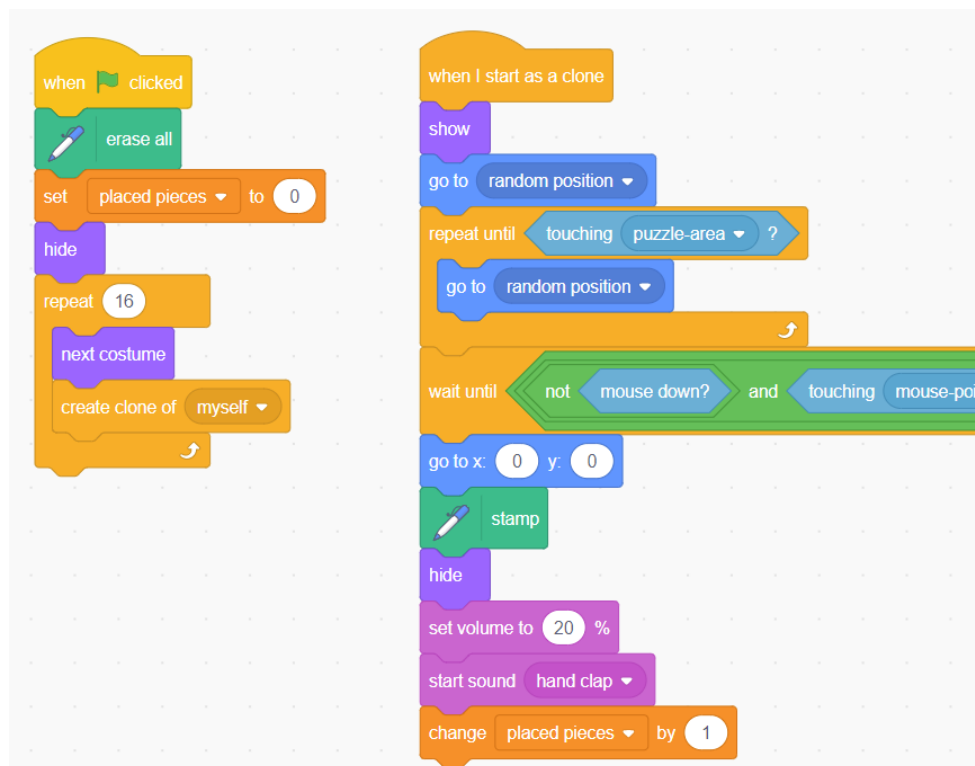
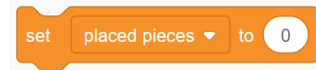


CHANGE VARIABLE BY ONE (00:52)

Let's add a block of code that adds 1 to the placed pieces variable each time a piece is placed. (Variable > **change placed pieces by 1**)



And another instruction in our first block of code that says: set placed pieces to 0 for when the game resets. (Variable > **set placed pieces to 0**)

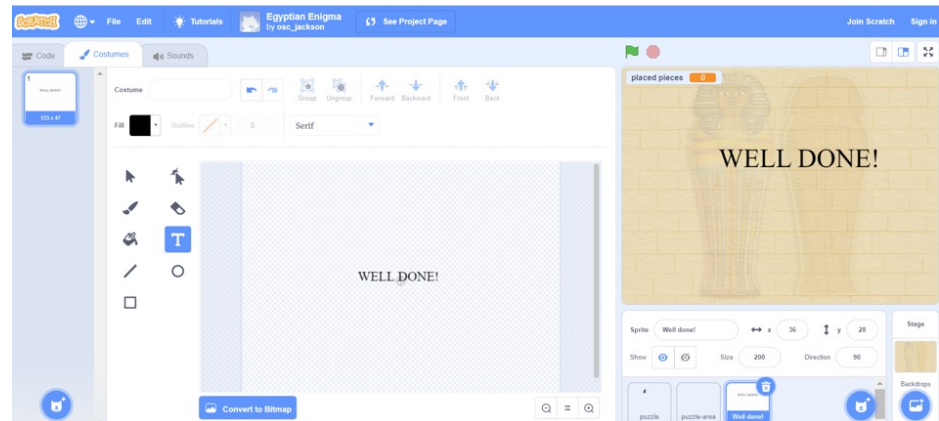
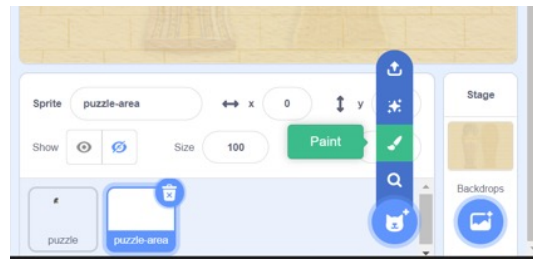


BROADCAST A MESSAGE (1:24)

Now let's add a block of code that says if there are sixteen placed pieces - **placed pieces = 16** - then to broadcast a message.



ADD YOUR MESSAGE (1:56)



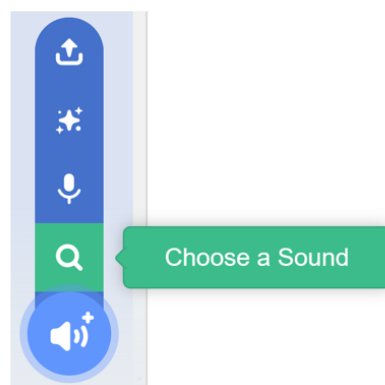
To add a message at the end of the game to congratulate the player, we are going to make a new sprite. I'm going to call mine "Well done"

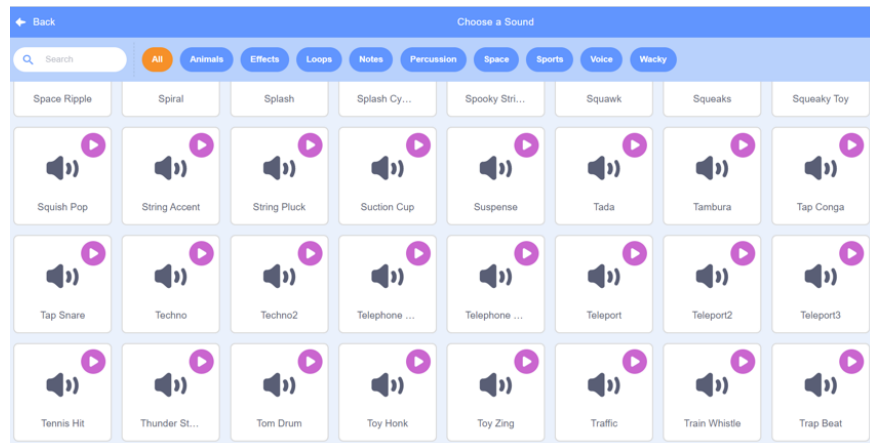
We are going to add two blocks of code to the Well done sprite.

The first one will make the sprite invisible when the flag is clicked to start the game
(When flag clicked_ hide)

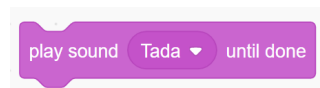
The second will receive broadcasts from the first sprite – the puzzle sprite. When it receives the message 1 broadcast, it will show the sprite, go to the front layer and glide up to the coordinates we set, (I'm going to use $x = -7$, $y = 127$) Make sure you start with it at the bottom of the stage. Let's also add a sound that plays with our message.

ADD A SOUND (3:30)

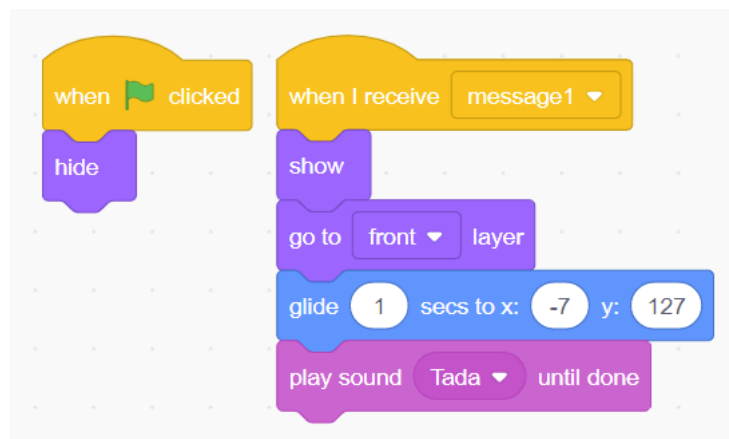




To add a sound, click on the Sounds tab, and on the **Choose a sound** icon. Select a sound from the menu. I'm going to choose 'Tada'. Now that you have added the sound to your list it will appear in the drop down menu of the Sound blocks. Add a sound block to the Well done sprite code that tells the program to play the sound 'Tada' until done.

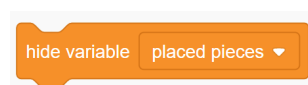


The code on the Well done sprite looks like this:



ONE LAST THING: MAKE THE VARIABLE INVISIBLE (4:14)

We don't really want to see the placed pieces variable on the stage when we're doing the puzzle, so let's add a block to hide the placed pieces variable on the first block of code for the puzzle sprite. (Variable > **hide variable placed pieces**)



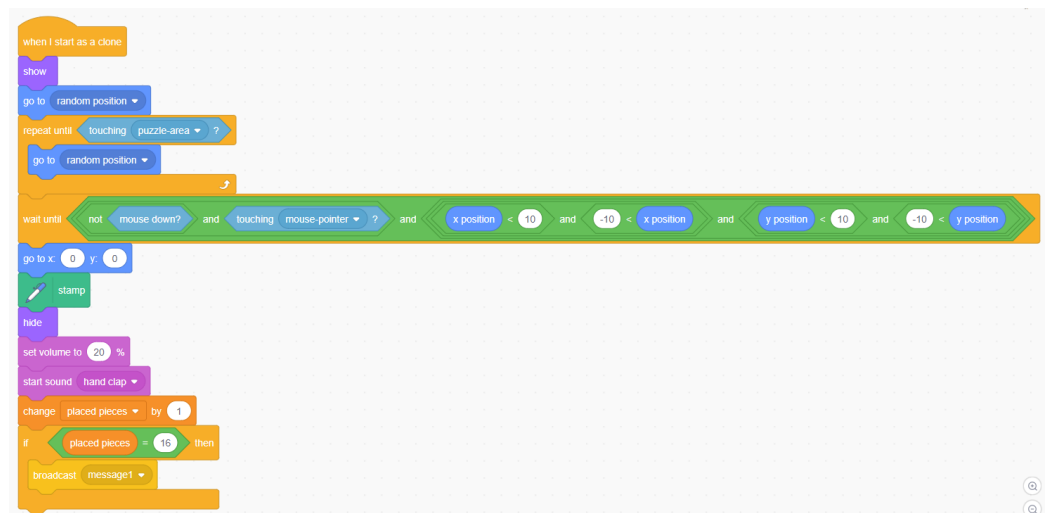
THE WHOLE CODE (4:44)

The whole of the code looks like this:

Puzzle piece sprite:

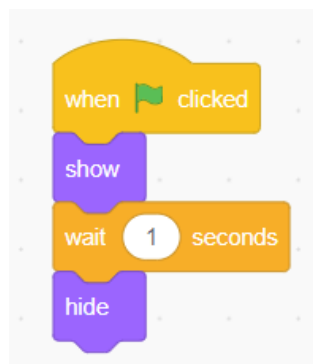


```
when clicked
  erase all
  hide variable placed pieces
  set placed pieces to 0
  hide
  repeat 16
    next costume
    create clone of myself
```



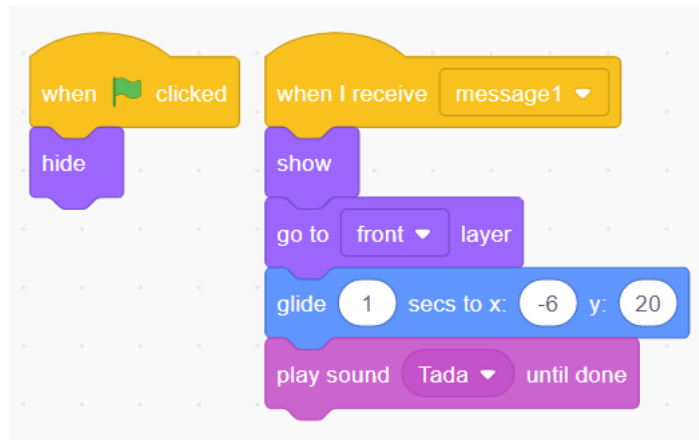
```
when I start as a clone
  show
  go to random position
  repeat until touching puzzle-area
    go to random position
  wait until not mouse down? and touching mouse-pointer? and x position < 10 and -10 < x position and y position < 10 and -10 < y position
  go to x: 0 y: 0
  stamp
  hide
  set volume to 20%
  start sound hand clap
  change placed pieces by 1
  if placed pieces = 16 then
    broadcast message1
```

Puzzle area sprite:



```
when clicked
  show
  wait 1 seconds
  hide
```

Well done sprite:



YOUR TURN (4:52)

Now that you know how this puzzle was made, you might like to have a go at creating a new puzzle of your own.

NGV SCHOOLS
PROGRAM PARTNERS



NGV DIGITAL CREATIVES IS
SUPPORTED BY PRINCIPAL PARTNER

