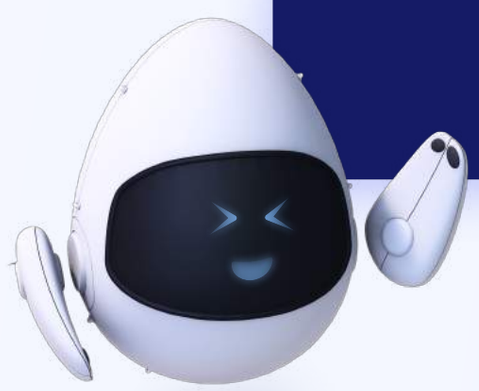
Guide For

Hatch Kid’s Blocks-Based Beginner Curriculum

Project 5:

VR Piano



Project 5:

VR Piano

Objective

In this guide we are going to explore some new concepts of programming like arrays,and conditional statements, and use these concepts to build a VR musical instrument that you can use to play music on your desktop/phones.

You wil also get to explore more about some basic 3D shapes, and X/Y/Z values for scale of an object, and learn to modify these properties using cursor events.

Concepts covered:

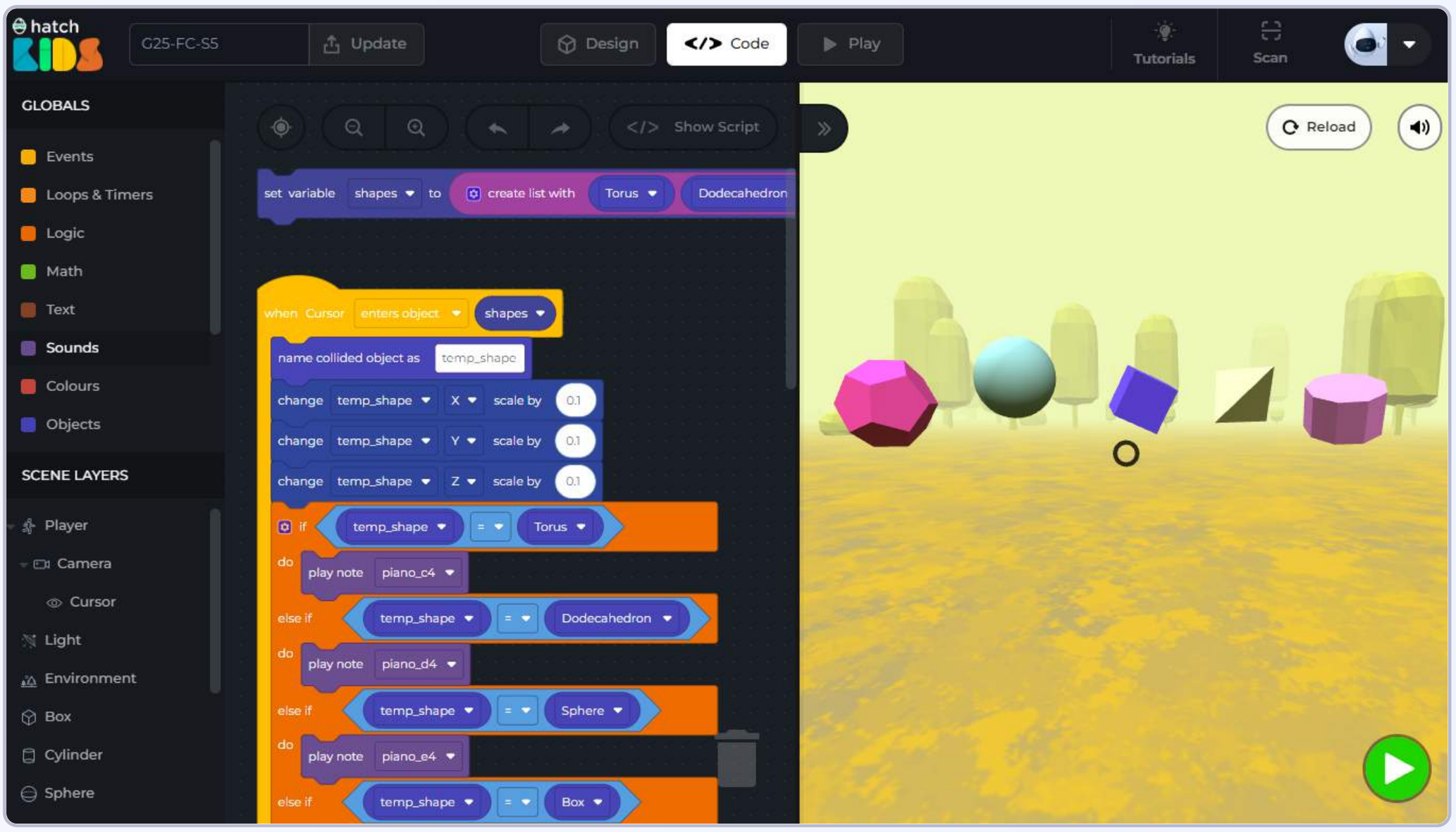
Basic 3D geometrical shapes
X/Y/Z values of scale of an object
Variables, arrays and their uses
If-else conditional statement

Final Output Link:

https://kids.hatchxr.com/@XR4schools/G25-FC-S5

Student Template Link:

https://kids.hatchxr.com/@XR4schools/G25-FC-S5-template

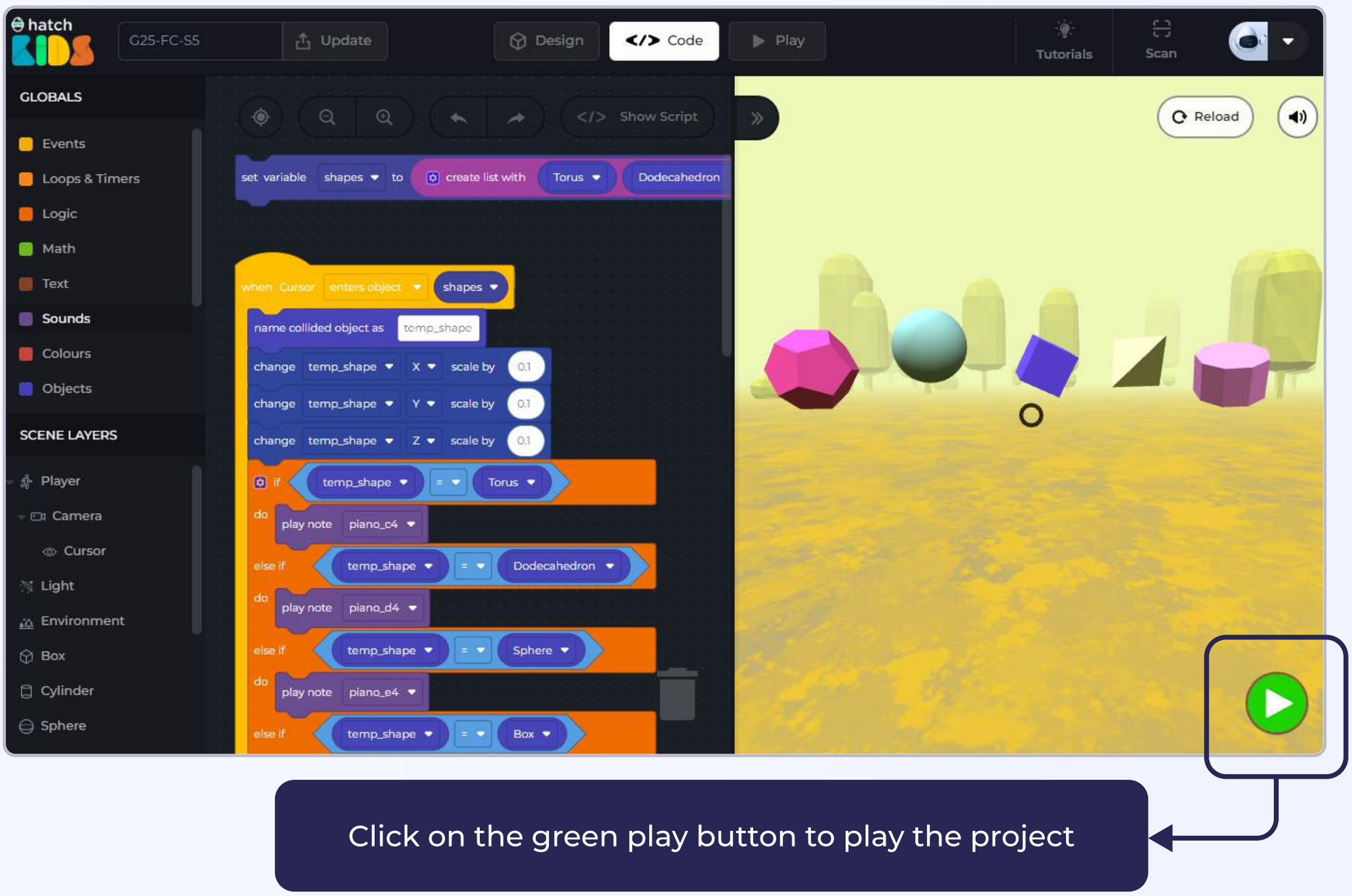


How it works?

Let’s first understand what we are going to be building in this session.

Open the completed project link: https://kids.hatchxr.com/@XR4schools/G25-FC-S5

You will a screen as shown below:



After you click on the **“Green Play Button”**, you may not notice anything happening in the game.

Move your mouse over the game screen, press and hold the left button of your mouse, and drag your mouse around the game.

There is a black circle at the center of the screen that moves with you. That black circle is called the **“cursor”,** and you can use it to interact with other 3D objects.

You will notice that whenever the cursor enters and of the 3D shapes present in the scene, the size of the shape increases, the color of the shape changes, and the sound of a musical note is played.

Also, as soon as the cursor leaves and of the 3D shapes in the game, the size of the object goes back to normal.

As you move the cursor over each of the text, and you will hear a different musical note play and if you move the cursor from left to right, passing through all the 3D shapes, it will sound lik you are playing a piano,

You will be learning how to build this project, and as you build this project, you will learn more about the cursor, variables, arrays and much more while building this.

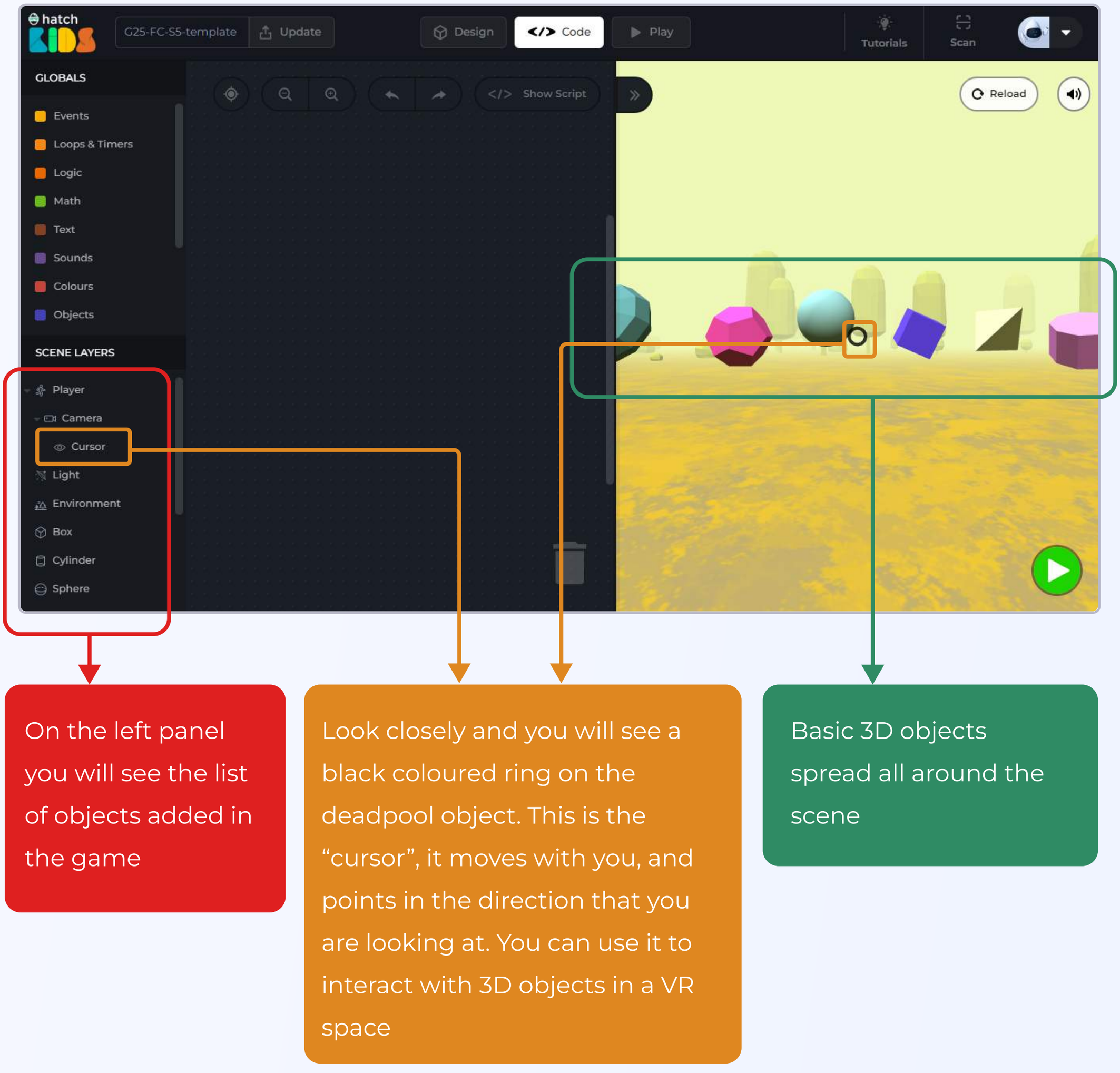
Let’s get started.

Objective 1: Understanding the template

Step 1: We will start by opening the template link of the project mentioned here.

Student Template Link: https://kids.hatchxr.com/@XR4schools/G25-FC-S5-template

Step 2: The link above will open up an empty project with no code in the hatch workspace, that looks as shown here. Let's understand the scene.



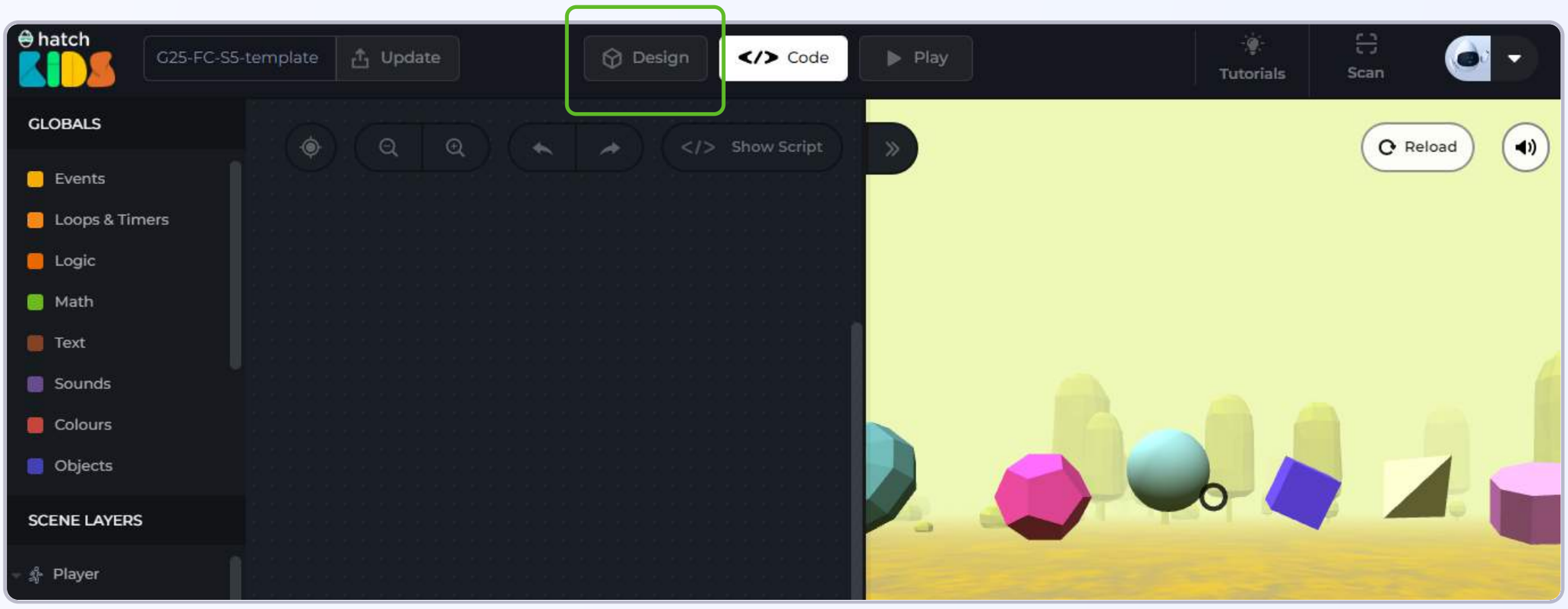
On the left panel you will see the list of objects added in the game

Look closely and you will see a black coloured ring on the deadpool object. This is the "cursor", it moves with you, and points in the direction that you are looking at. You can use it to interact with 3D objects in a VR space

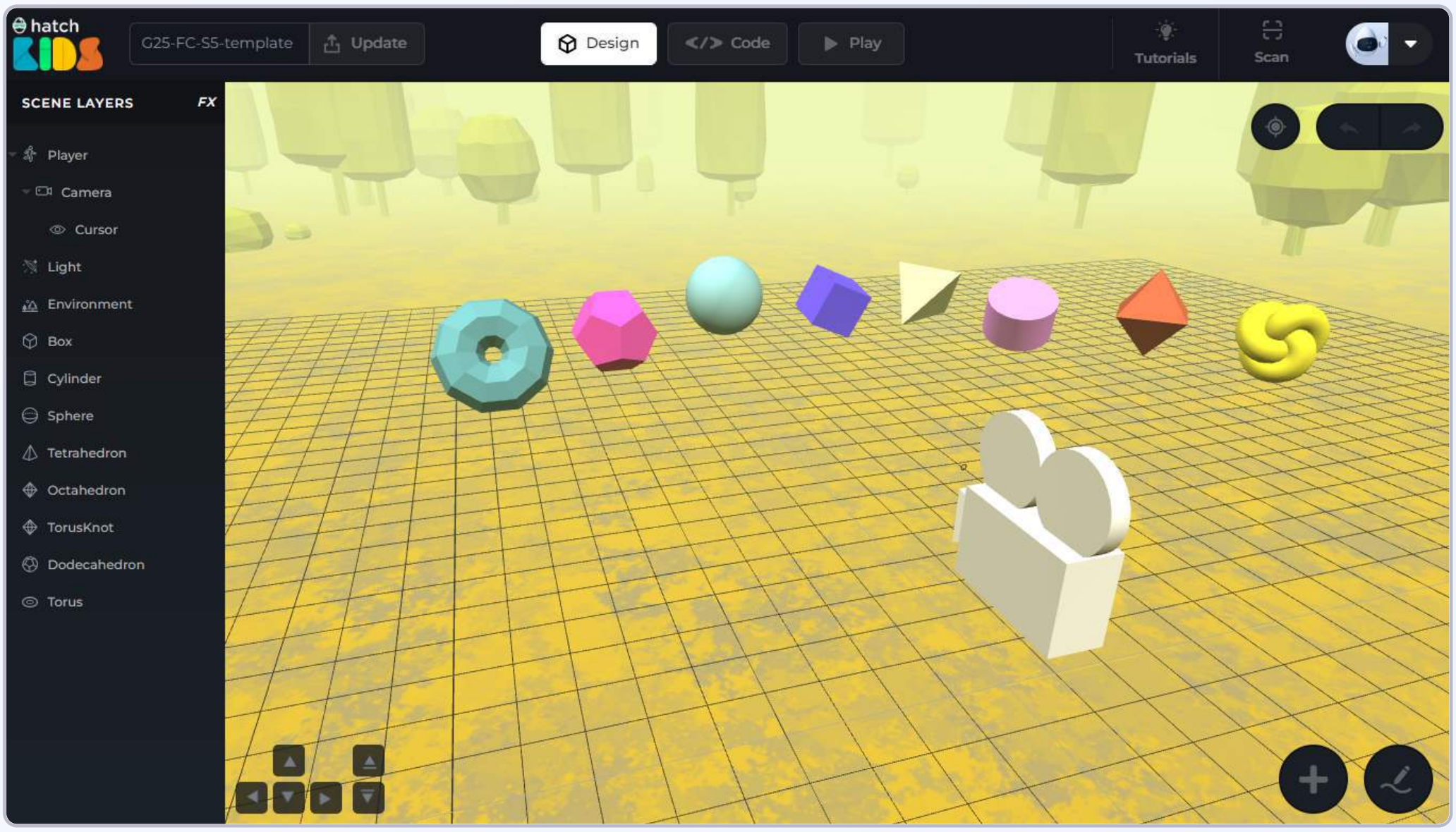
Basic 3D objects spread all around the scene

As you can see there are lots of simple 3D shapes in this project. Let's have a look at each of those shapes, and understand the design of this project.

Step 3: Click on the design tab at the top of the workspace.

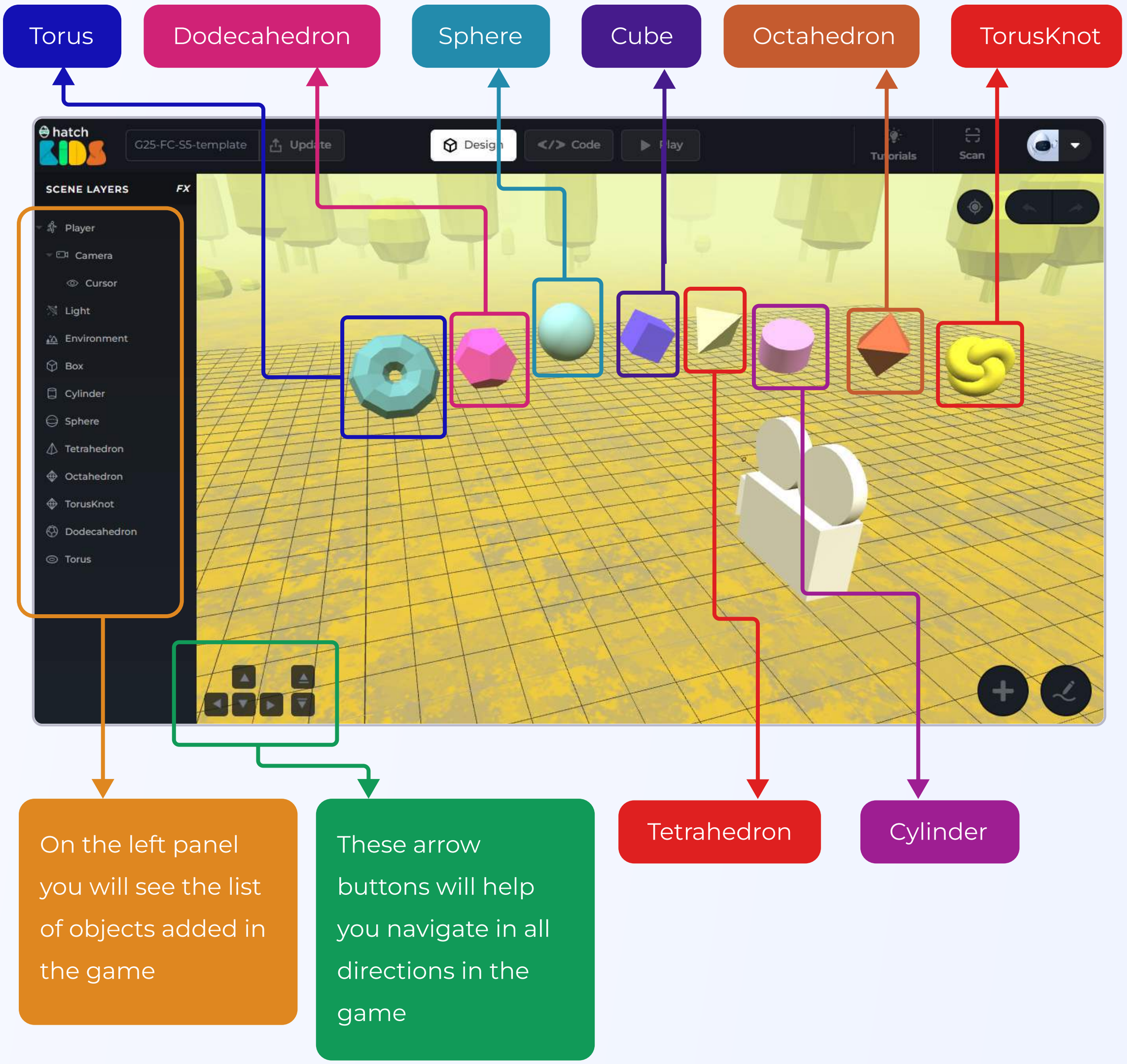


You will switch to the design tab of the hatch workpace, and your screen would look as shown:



This is where you can design your own game, add any 3d models that you want in the game and make the game look like whatever you want.

Here the design of the game have already been prepared, and you are learning how to code the games. We are going to element the code for cursor to interact with each ot these objects, which is why it's important for you to know about the different 3D shapes that you can see in this project.



In the design tab, you will see a line of simple 3D shapes. From left to right the objects are:
Torus, Dodecahedron, Sphere, Cube, Tetrahedron, Cylinder, Octahedron, TorusKnot

You can see the names of these objects in displayed in the left panel as well.

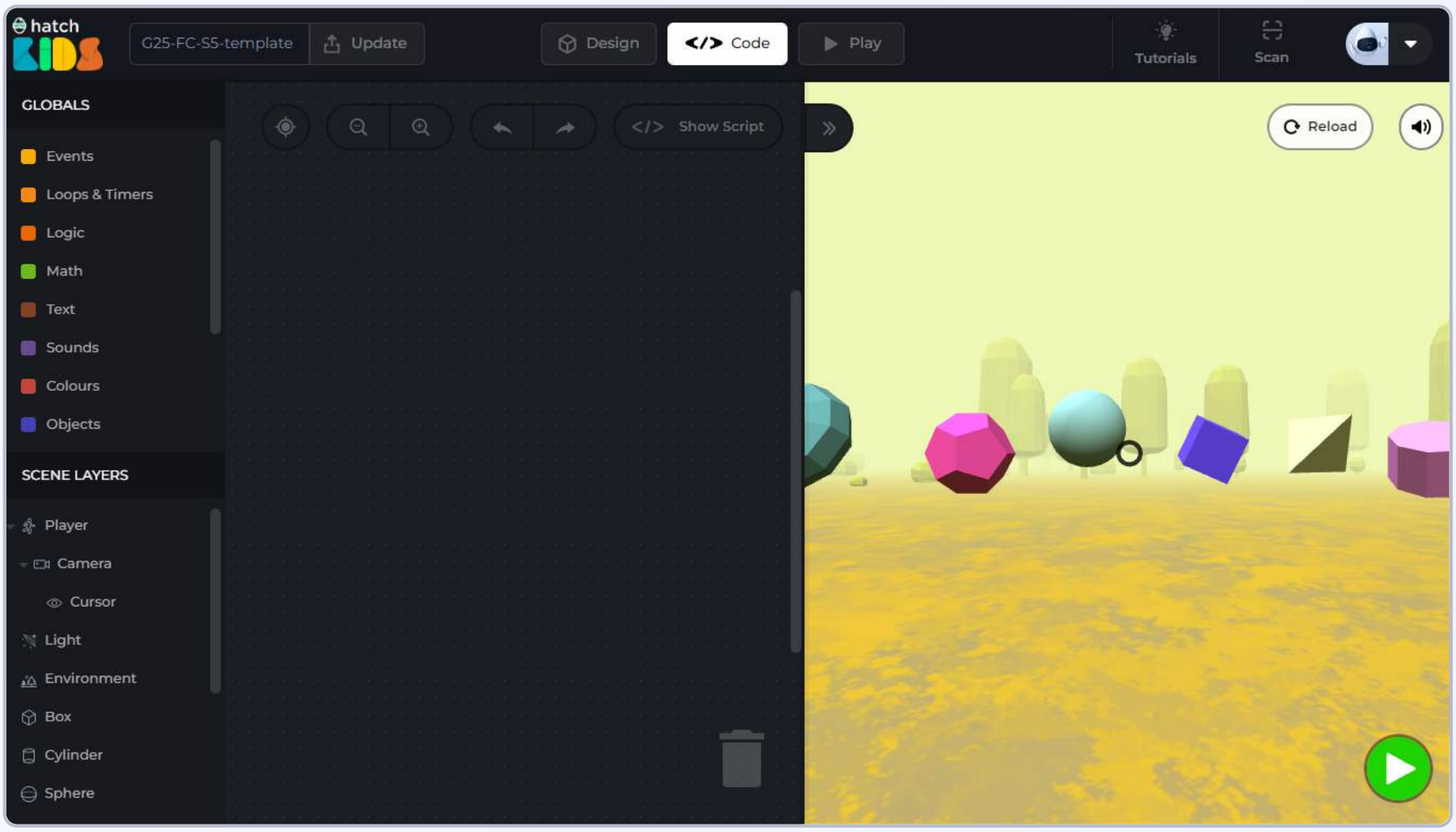
There are 8 shapes in the game, and we are going to use these 8 objects to play eight musical notes (one octave) of a piano.

We are going to use cursor interaction, as cursor enters each of the shapes here, the size of the shape will increase an dit will play a musical note.

Let’s start coding.

Objective 2: Changing size of an Object

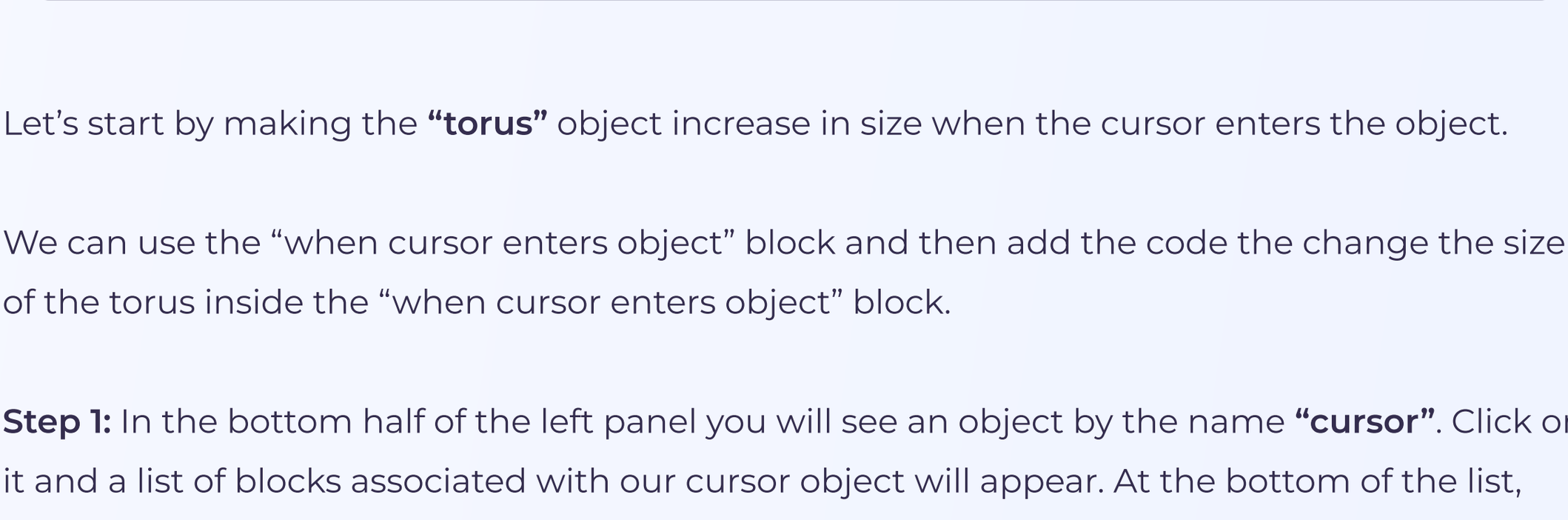
Step 0: Click on the code tab at the top of the screen and switch back to the code window.



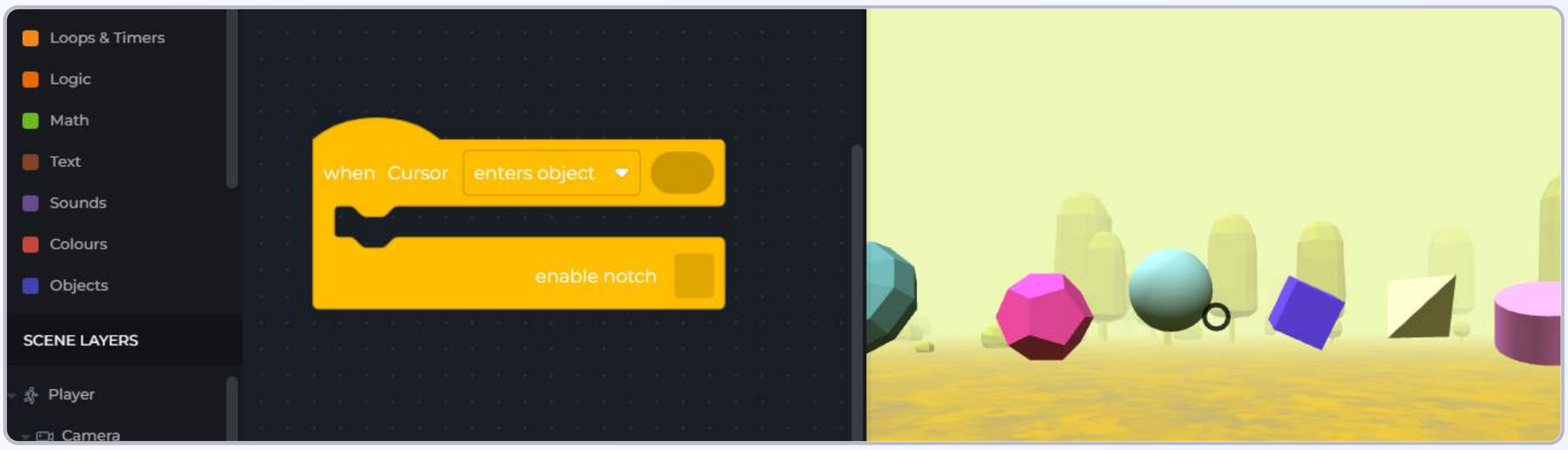
Let’s start by making the **“torus”** object increase in size when the cursor enters the object.

We can use the “when cursor enters object” block and then add the code the change the size of the torus inside the “when cursor enters object” block.

Step 1: In the bottom half of the left panel you will see an object by the name **“cursor”**. Click on it and a list of blocks associated with our cursor object will appear. At the bottom of the list, there will be a block called **“when cursor enters object”**



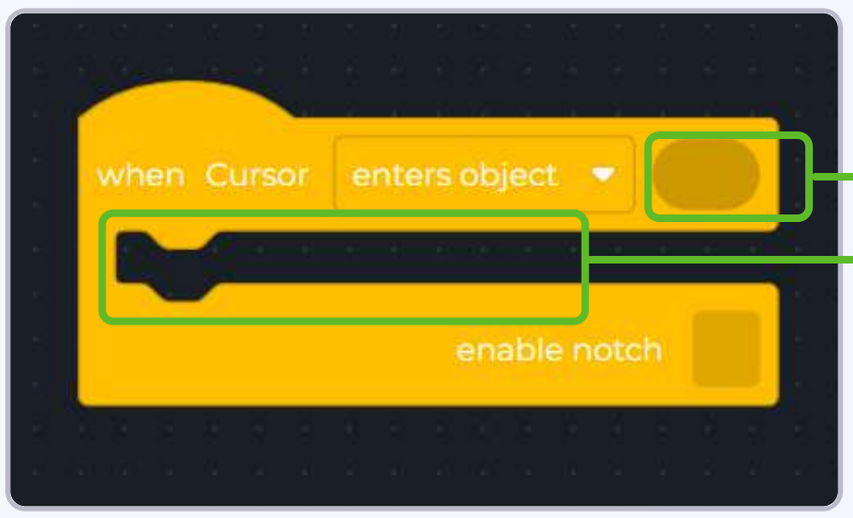
Step 2: Click and drag the **“when cursor enters object”** block into the workspace.



We know that in a 3D scene with multiple objects, **the cursor can:**

1. enter an object
2. leave an object (after entering that object)
3. click on an object (while its inside an object)

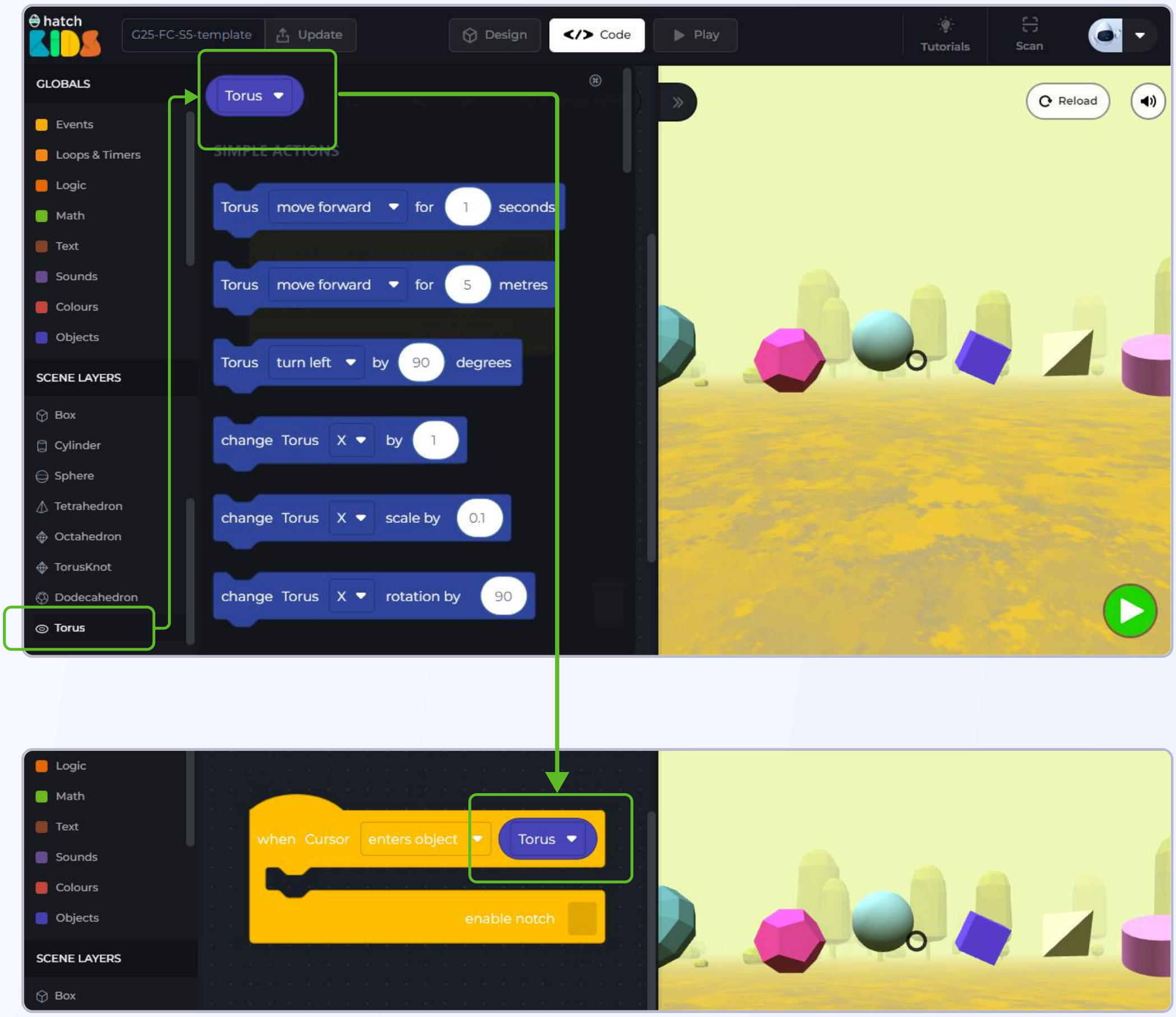
Inside the **“when cursor enters object”** block, we can define the name of the object with which we want the cursor to interact.



Step 3: In the bottom half of the left panel, find the name **“Torus”**. That is the name of the shape placed to the extreme left of the scene.

Click on the name “Torus” in the left panel and at the top of the list of block that appears, you will see a block that only says, “Torus”

Click and drag the **“Torus”** block and place it inside the **“when cursor enters object”** block as shown.



Now we need to define what needs to happen in the game when the cursor enters the torus object.

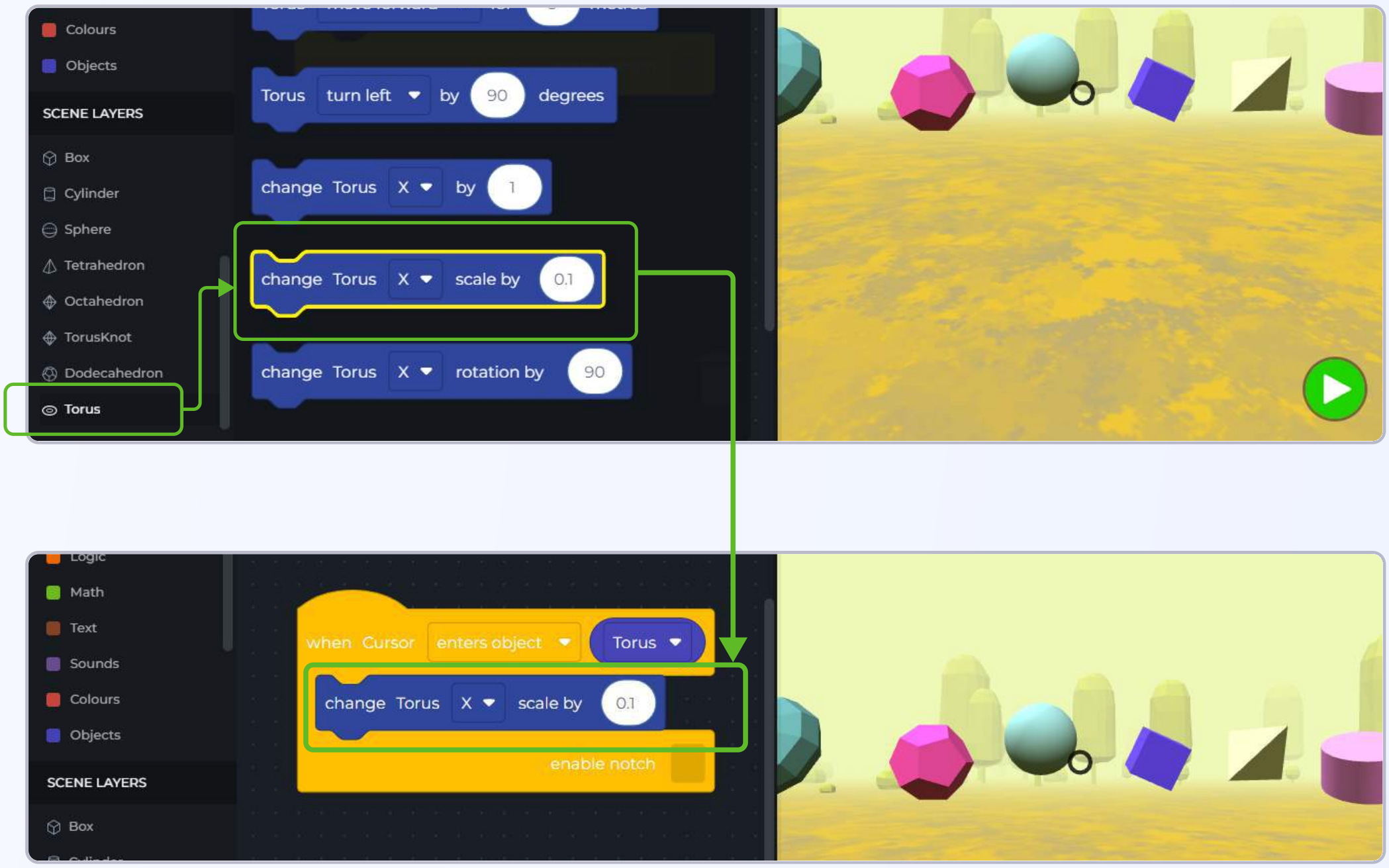
We want to increase the overall size of the torus when cursor enters it.
Just like position has X/Y/Z values, similarly, for size we have X?Y/Z scale values of an object.
The X/Y/Z scale values determine how big or small an object is going to be.

To increase the overall size of an object, we need to change all the X, Y, and Z scale values together, Or you can chose to just change one of X/Y/Z scale values, thereby making the object wider or taller as per your requirements.

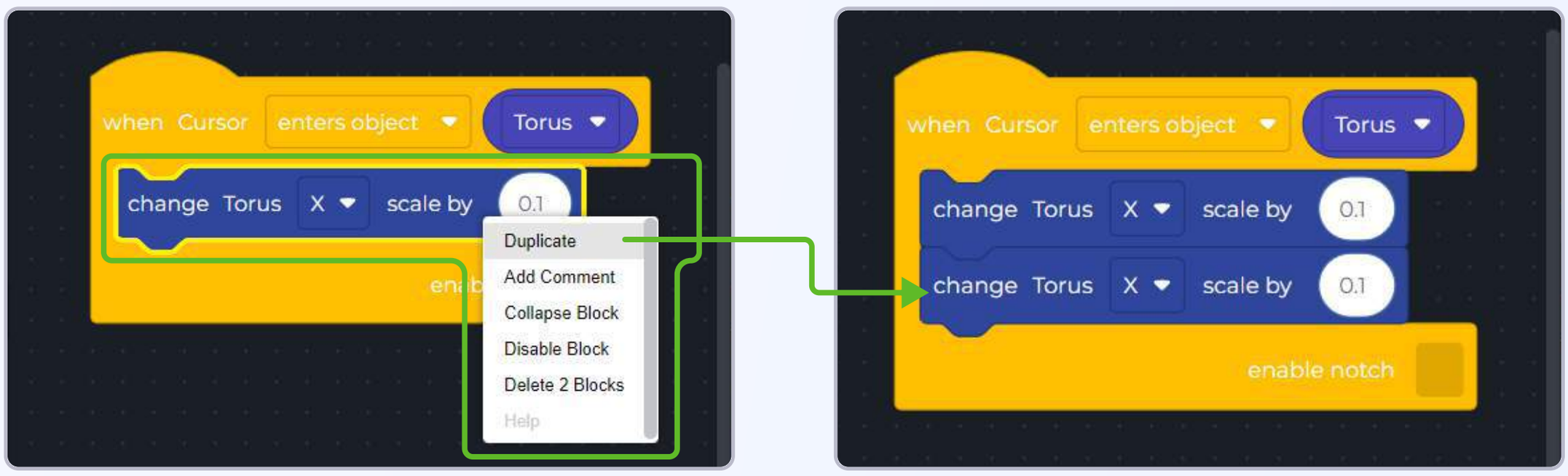
For now, let’s increase the overall size of the torus object.

Step 4: Since we want to change the size of the torus object, so click on the name **“Torus”** on the left panel. In the list of blocks you will find a block that says, **“change Torus X scale by 0.1”.**

Click and drag out the **“change torus X scale by 0.1”** block, and attach it inside the **“when cursor enters object torus”** block.



Step 5: Right click and duplicate the “change Torus X scale by 0.1” block. A new “change Torus X scale by 0.1” block will appear on the screen. Attach this new block inside the “when cursor enters” block as well

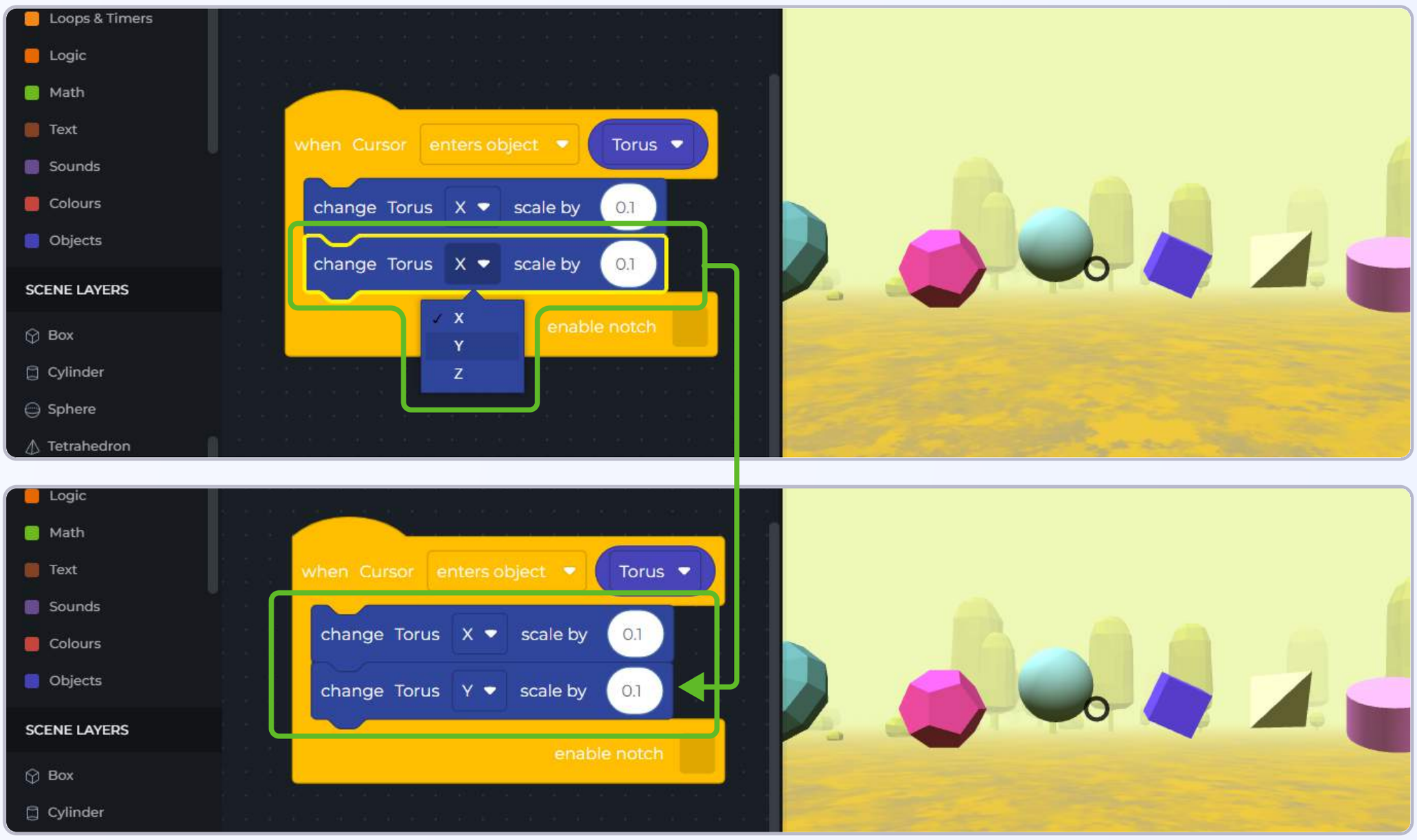


Step 6: Click on the **drop-down menu** option around the character **“X”** in the second “change Torus X scale” block.

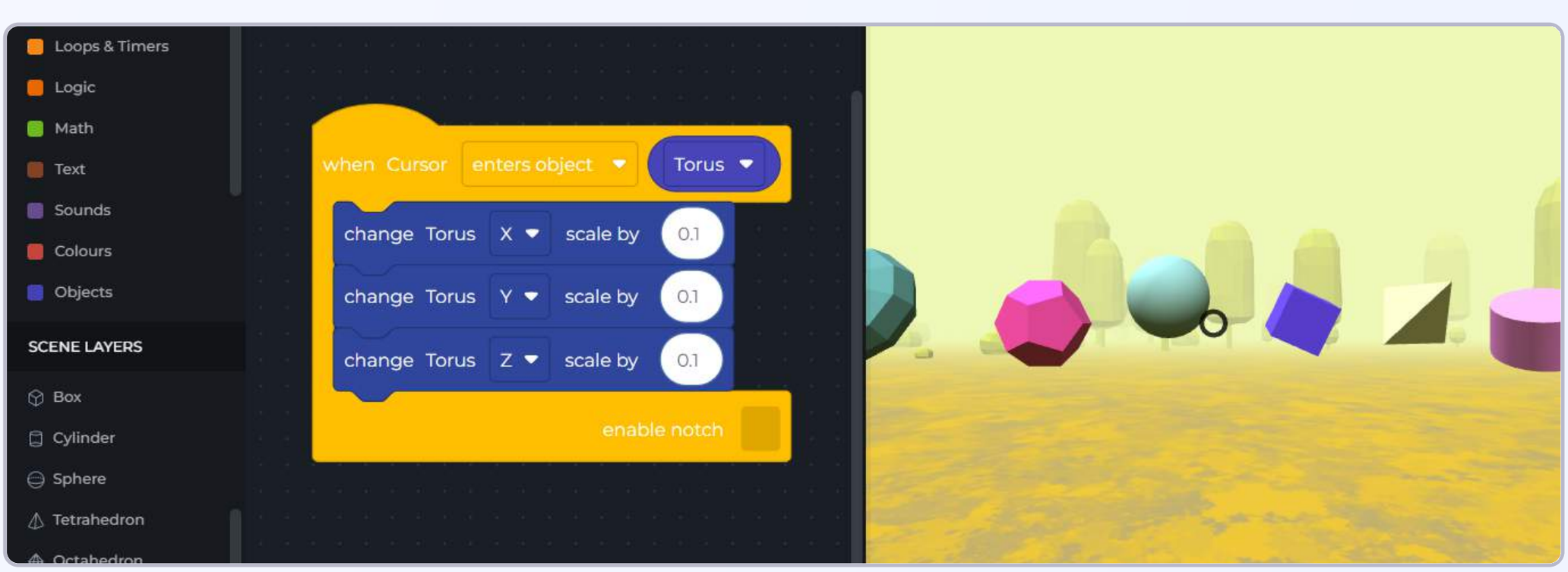
You will see a **list of options** displaying: **“X”, “Y”, and “Z”.**

Select the option “Y”.

Your new block would now say **“change Torus Y scale by 0.1”**



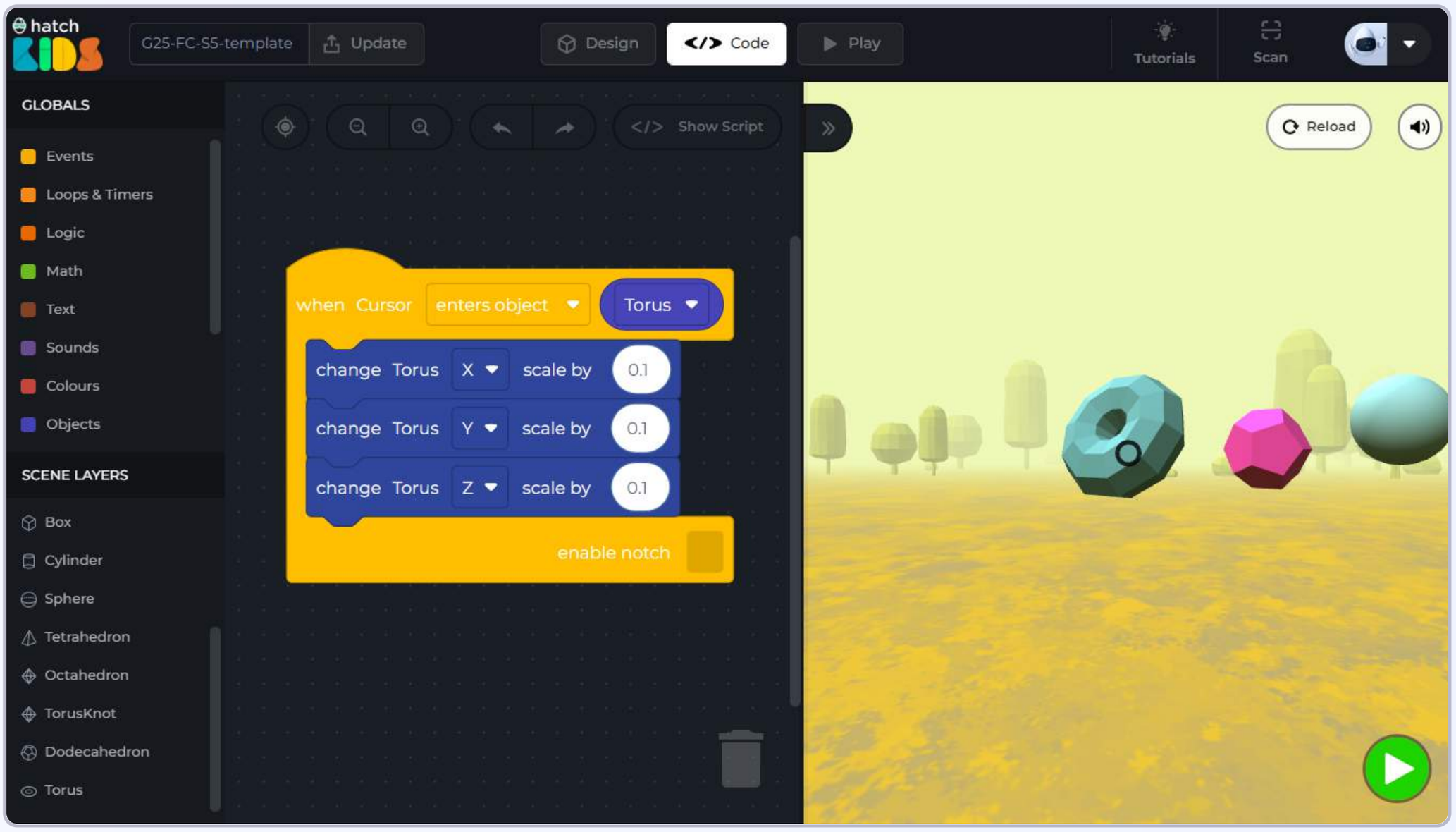
Step 7: Duplicate the “change torus X scale by 0.1” block once again, and this time change the “X” scale to “Z” scale.



Thus, through the blocks, we are telling our computers that, anytime the cursor moves over the “Torus”object, the size of the torus object should increase by 0.1 units in al directions.

Click on the green play button to run this code, and then drag your cursor (black ring at the center of the game) over to the torus object on the extreme left of all the objects.

You will see the size of the torus increase. In fact, every time the cursor moves over the torus object, its size will increase.



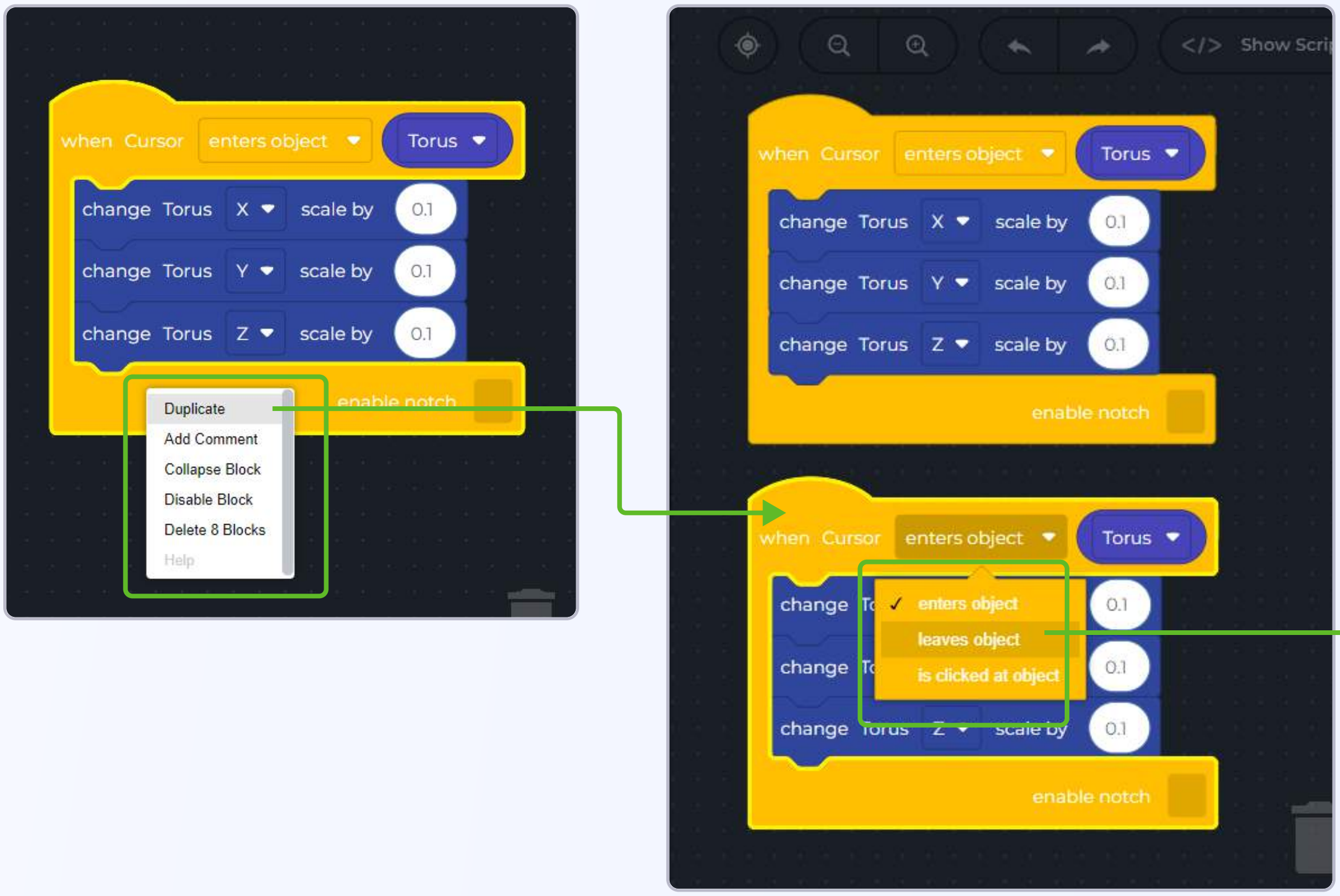
Let’s write the code such that, when the cursor enters the torus object, its size increases, and when the cursor leaves the torus object, its size decreases and goes back to normal.

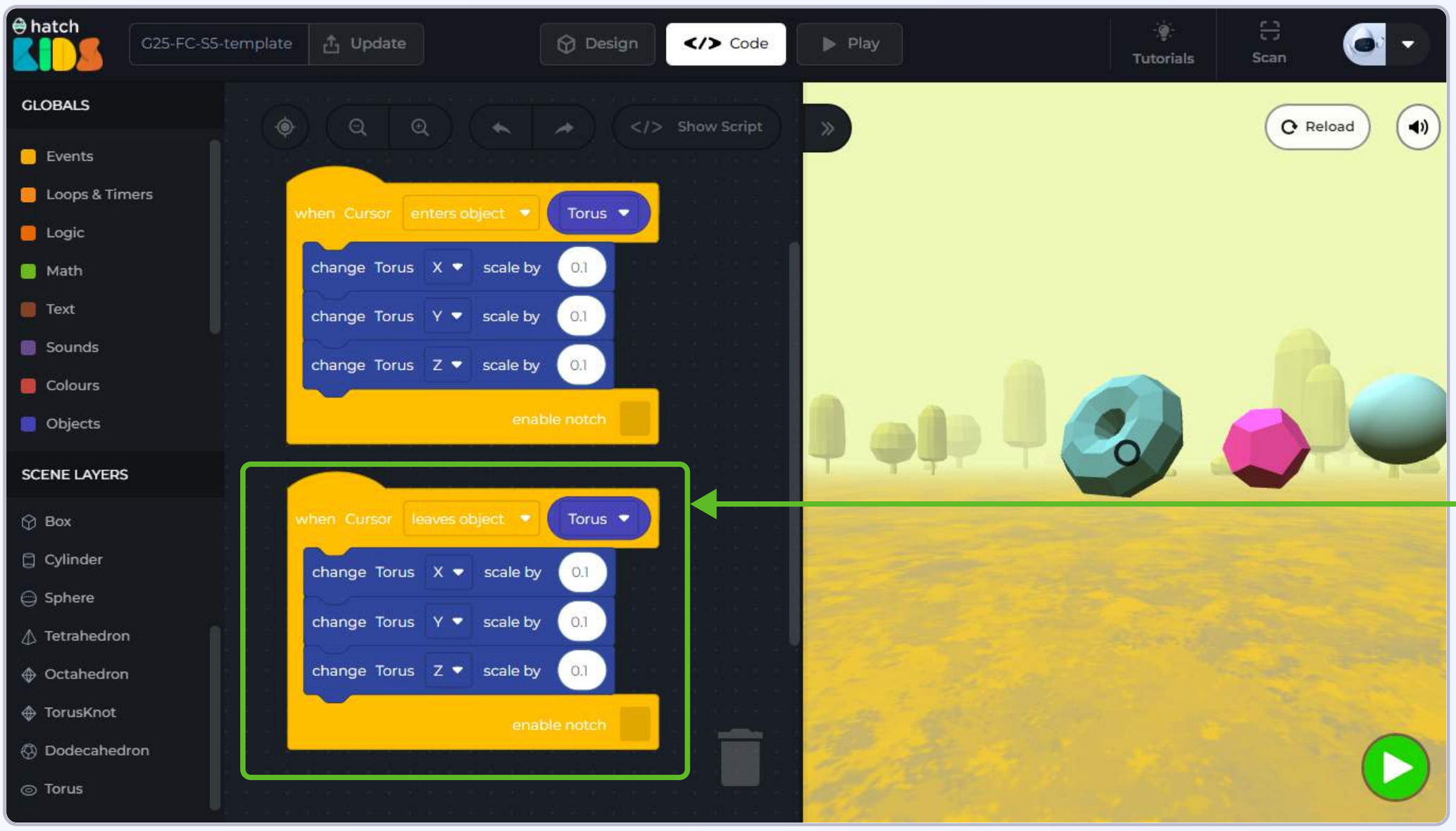
To do this, first click on reload and reset your game and then:

Step 8: Right click on the **“when cursor enters object Torus”** block and select **“duplicate”** to **create a copy** of the entire block.

In the new **“when cursor enters object”** block, **click on the “drop down option”** around the **“enters object”.** You will see a list with the options:

- **enters object**
- **leaves object**
- **is clicked at object.**


when Cursor enters object Torus
change Torus X scale by 0.1
change Torus Y scale by 0.1
change Torus Z scale by 0.1
enable notch
Duplicate
Add Comment
Collapse Block
Disable Block
Delete 8 Blocks
Help
Show Scri
when Cursor enters object Torus
change Torus X scale by 0.1
change Torus Y scale by 0.1
change Torus Z scale by 0.1
enable notch
when Cursor enters object Torus
enters object
leaves object
is clicked at object
0.1
0.1
0.1
enable notch

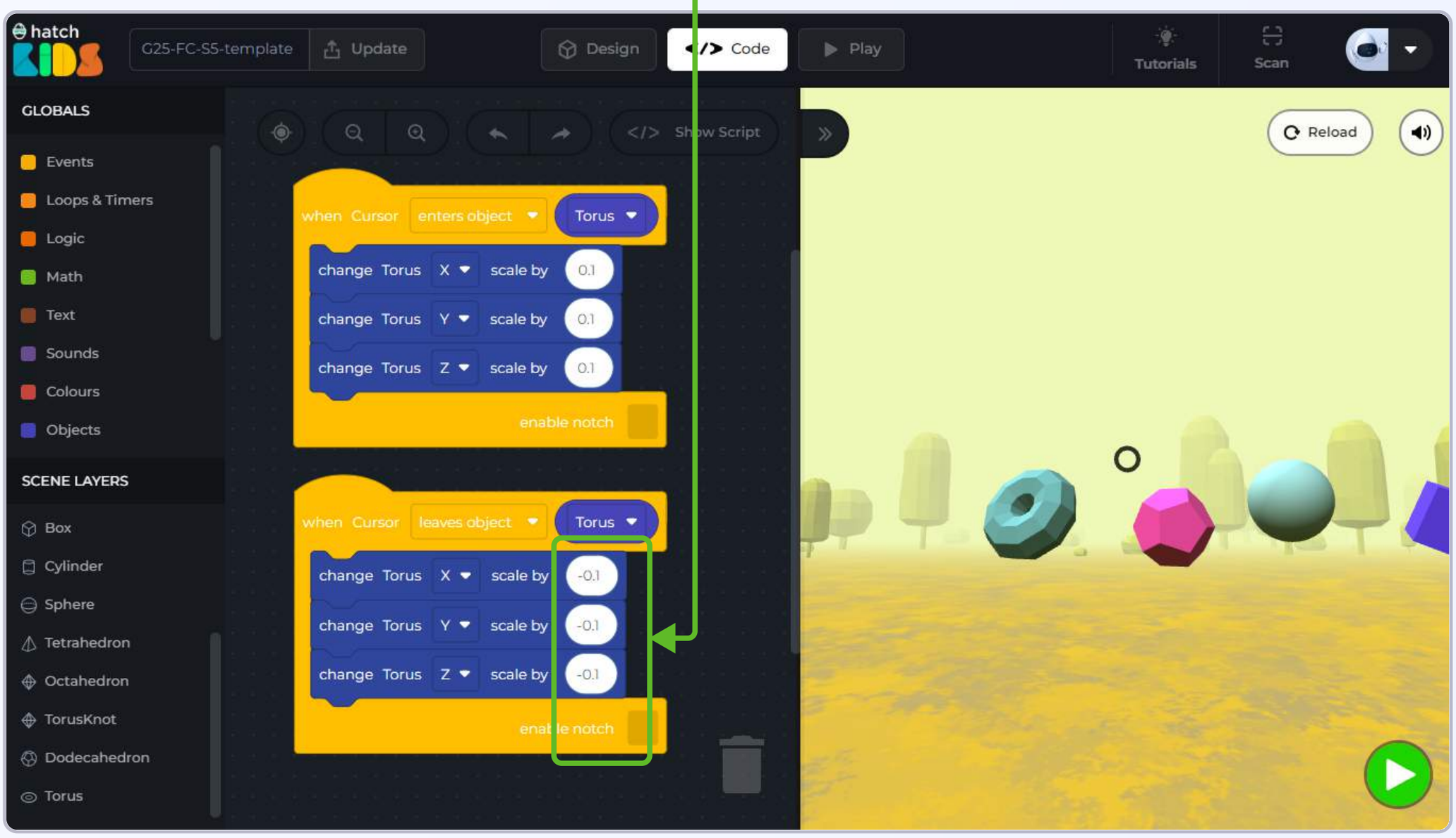

hatch KIDS
G25-FC-S5-template
Update
Design
Code
Play
Tutorials
Scan
GLOBALS
Events
Loops & Timers
Logic
Math
Text
Sounds
Colours
Objects
SCENE LAYERS
Box
Cylinder
Sphere
Tetrahedron
Octahedron
TorusKnot
Dodecahedron
Torus
Show Script
when Cursor enters object Torus
change Torus X scale by 0.1
change Torus Y scale by 0.1
change Torus Z scale by 0.1
enable notch
when Cursor leaves object Torus
change Torus X scale by 0.1
change Torus Y scale by 0.1
change Torus Z scale by 0.1
enable notch
Reload

We want to decrease the size of the torus back to normal when the cursor is not inside the torus object.

So mathematically, if when cursor moves inside the torus, we increase the scale in X?Y/Z by 0.1 Then similarly, when cursor leaves the object, we can decrease the scale in X/Y/Z by 0.1.

Decreasing the value of X/Y/Z scale by 0.1 is same as change the value by “-0.1”

Step 9: In the **“when cursor leaves object Torus”** block, **change the value** inside all the number blocks **from “0.1” to “-0.1”**



Click on the green play button to run the new code, and you will see, that as you move the cursor inside the torus object, its size increases, and as you move the cursor outside of the torus object, its size decreases and goes back to normal again.

🎉🎉🎉 **CONGRATS!! Youjust completed the first section of this project** 🎉🎉🎉

Objective 3: Understanding LIST blocks

You coded a simple cursor interaction with the Torus object.

We have to code the similar interaction of cursor with all the other shapes in the game.

One way to do this would be to duplicate the “cursor enters object” and “cursor leaves object” blocks 8 times for all the 8 objects, and then add the blocks to change the scale of each of those objects inside the cursor blocks.

The code that you may end up writing may work as needed but clearly this is not the best way to write this code.

We want to perform the same activity of increasing and decreasing the size of an object -- and this object would be one of the 8 shapes present in the scene, based on where the cursor is present.

In situations like these, we can use **“list” blocks.**

A list is just another variable, that can store multiple values inside of it at the same time.

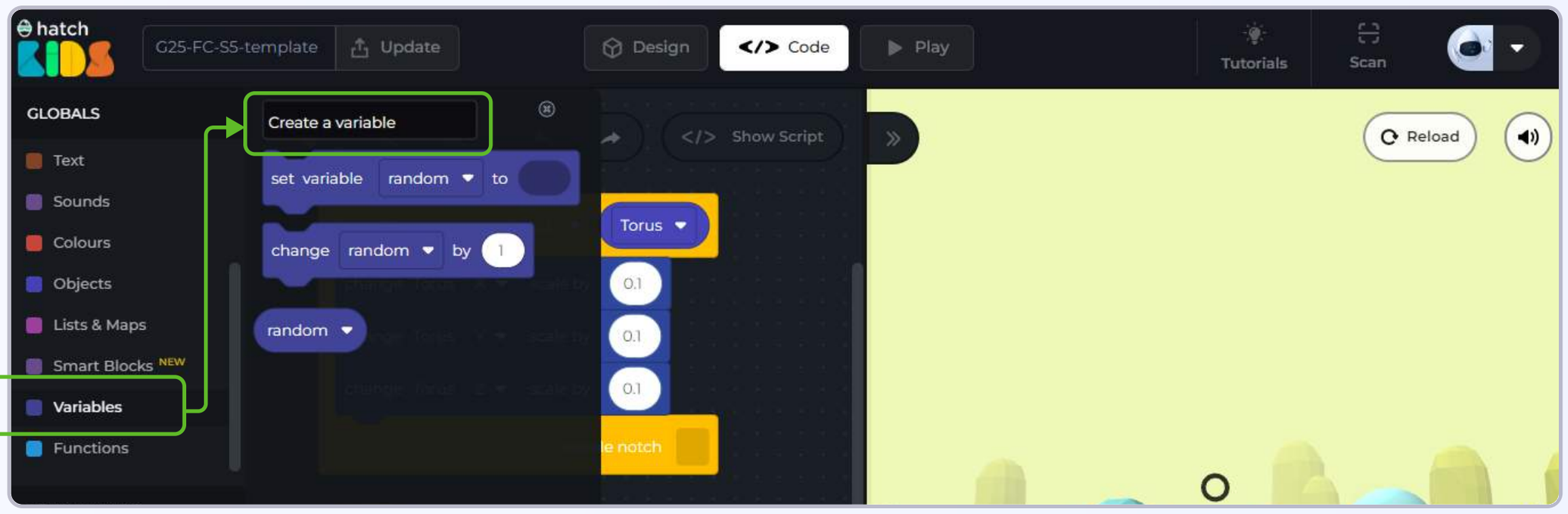
We can create a list of all these 8 shapes and save it in a variable.
Then we can use the same “when cursor enter/leaves object” block, and code the logic to say that when cursor enters/leaves any one of the eight shapes mentioned in the list, then increase/decrease the size of that shape.

Let’s try it out.

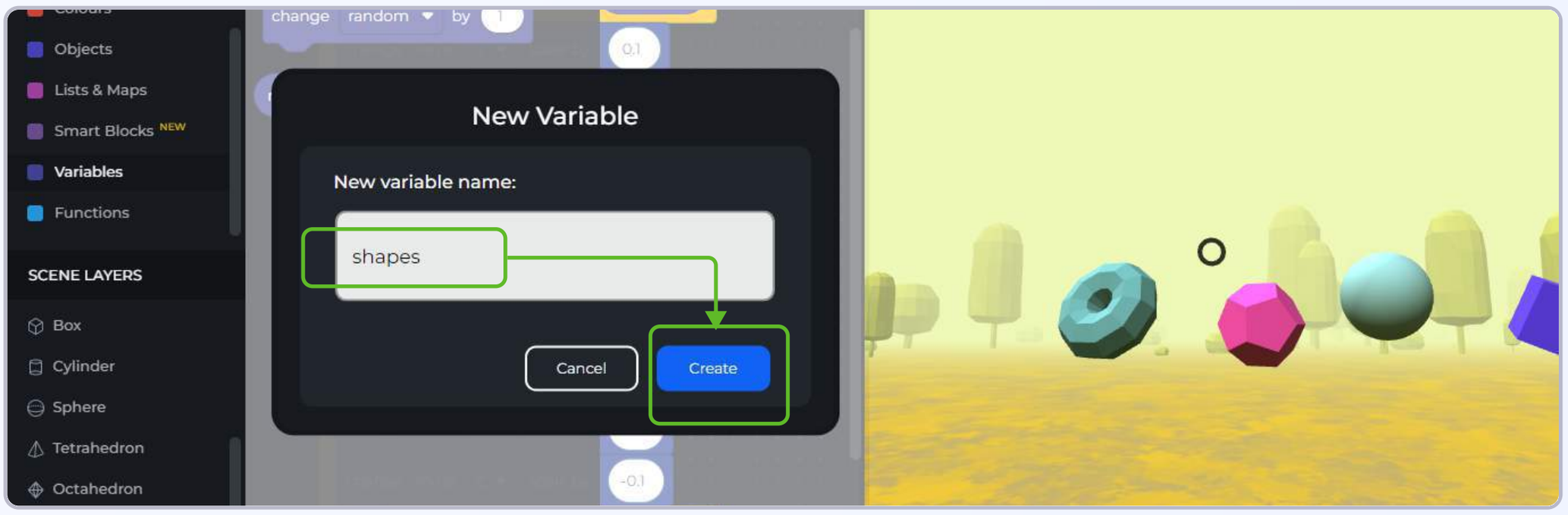
1. We will start by creating a variable called “shapes”
2. Then in this “shapes” variable we will need to store a list of 8 shapes present in our scene
3. We will need to modify the “when cursor enter/leaves object” block to work for all shapes in the list that we create.

Then we can add sounds for each of the shapes to make it sound like we are playing a piano.

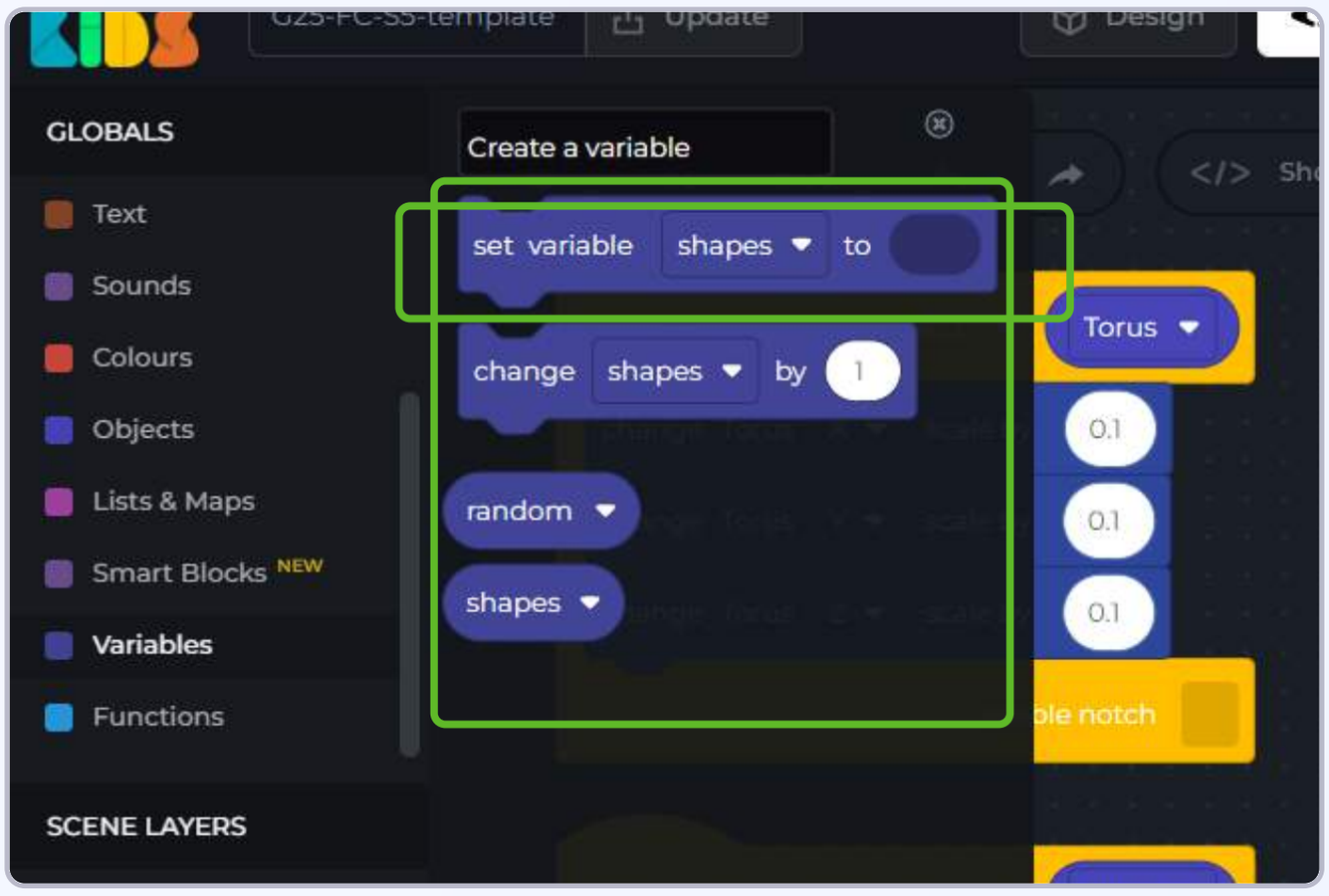
Step 1: Click on the **“variables”** section in the left panel, and click on the button that says **“create a variable”**



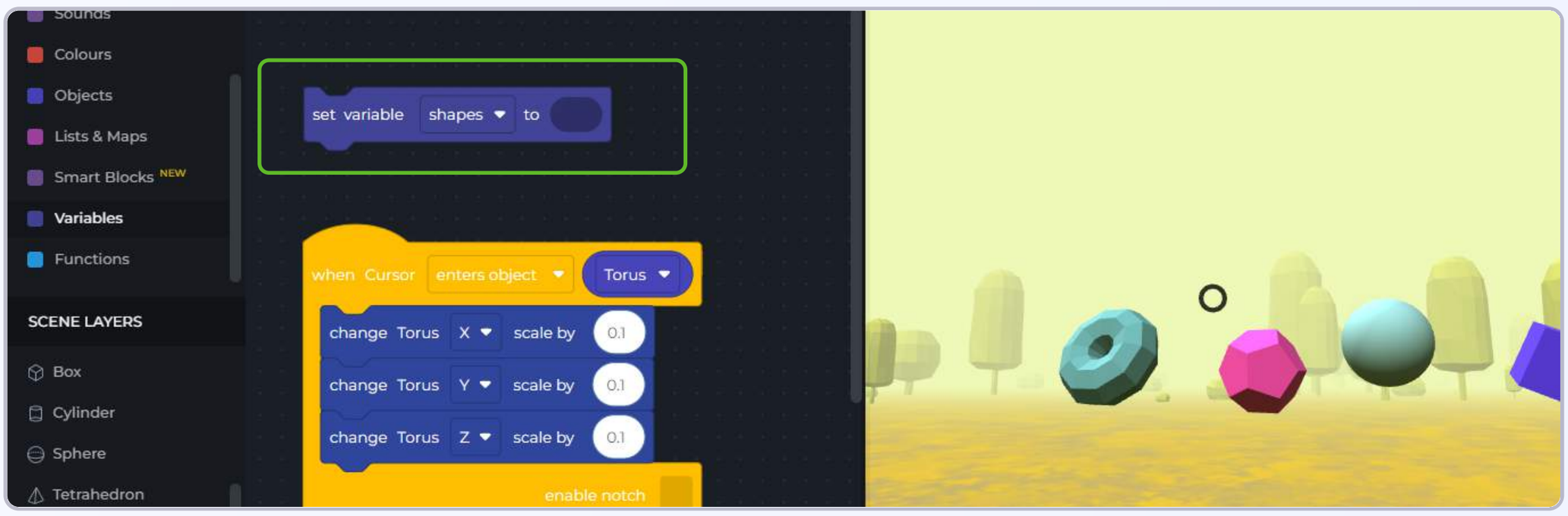
Step 2: A window will appear asking you to give a name to your variable. Let’s call this variable **“shapes”** and click on the **“create”** button



Once you click on the “create” button, in the variables section you will see the following blocks appear.

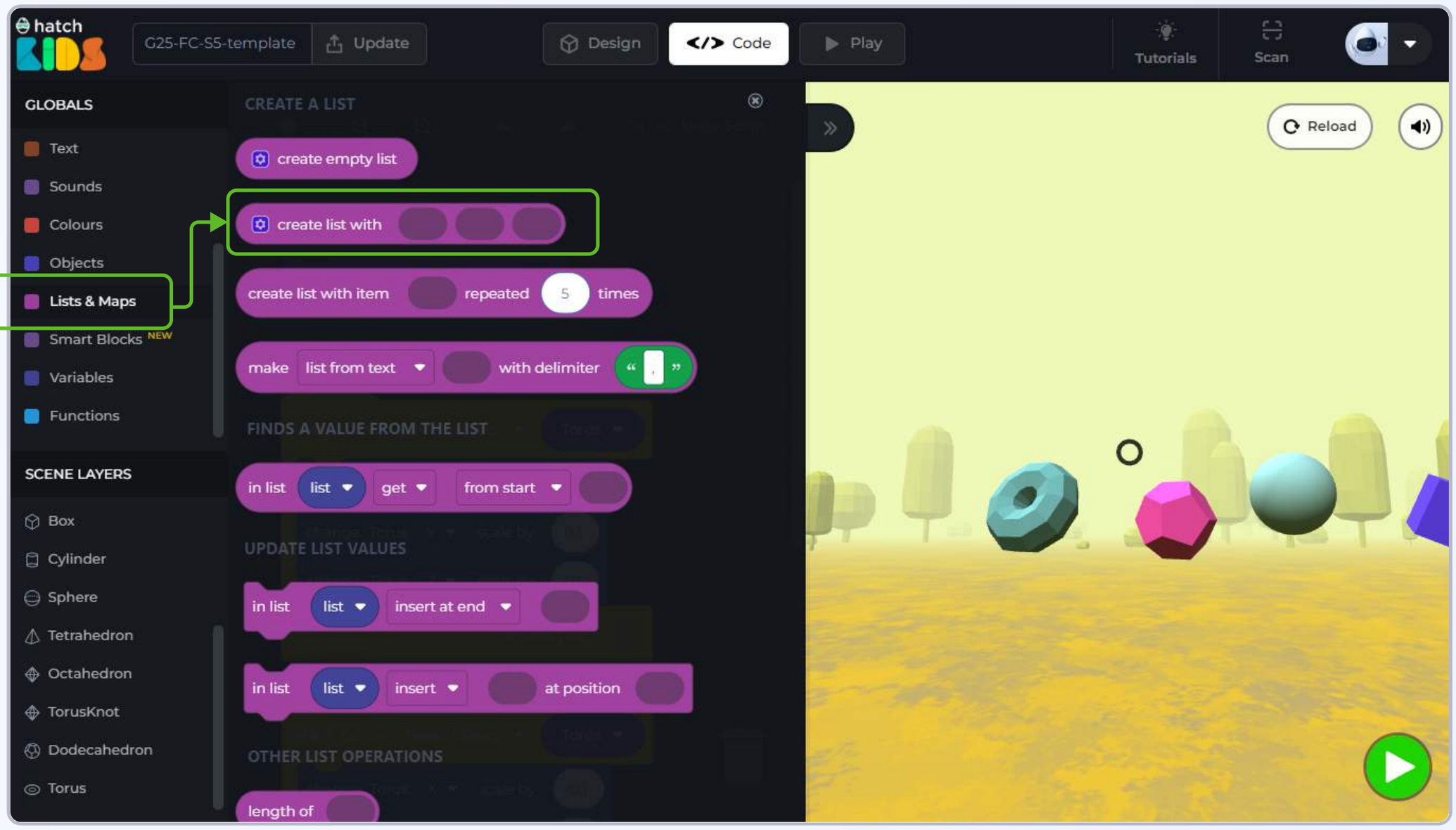


Step 3: Drag out the **“set variable shapes to”** block inside the workspce.

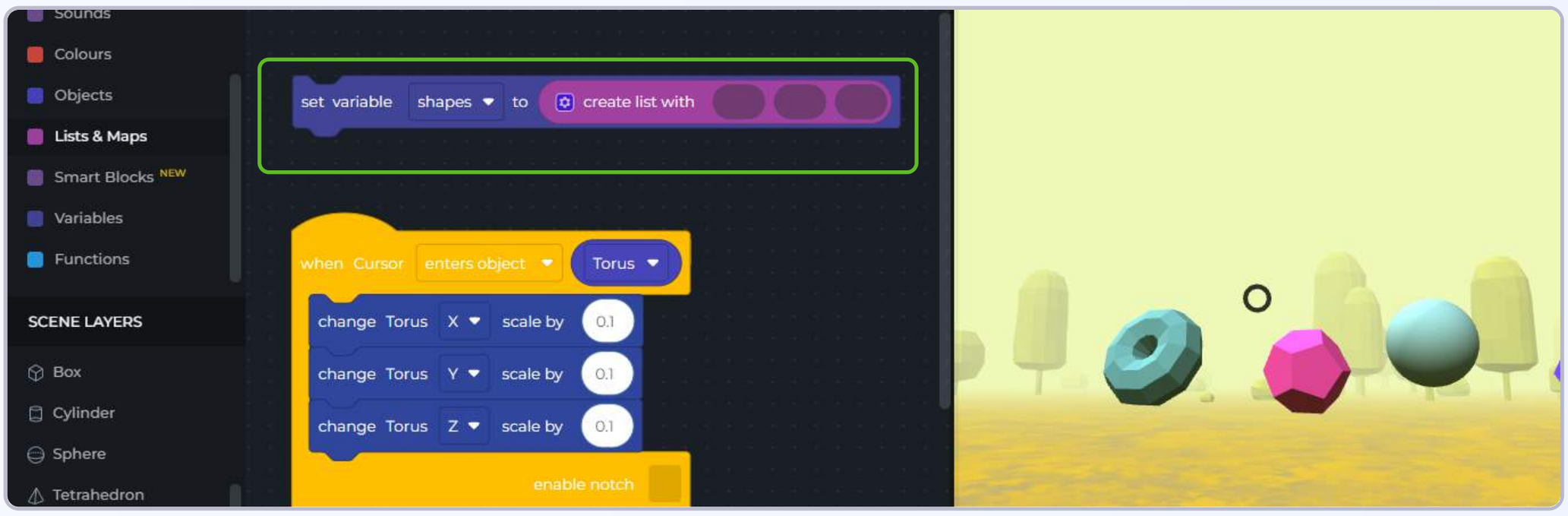


Let’s now make a list of all the eight objects and save it in the shapes variable

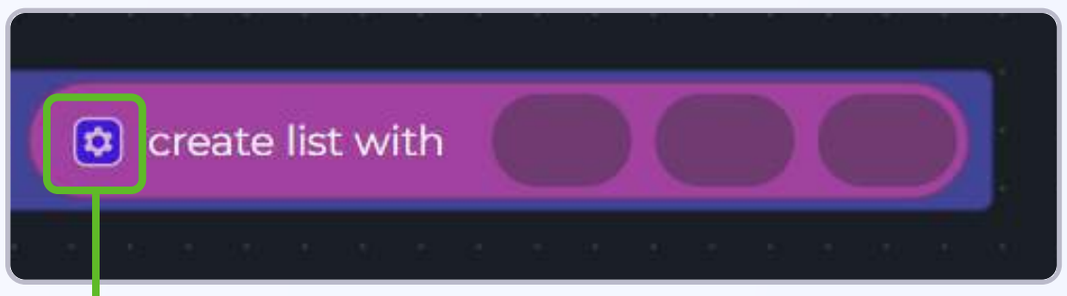
Step 4: In the top half of the left panel of your workspace, there is a category of blocks called **“lists & maps”**. Click on it, and drag out the block that says **“create list with”**



Step 5: Drag out the **“create list with”** block in the workspace, and attach it inside the **“set variable shapes to”** block.



As you can see the “create list with” block has 3 empty spaces. This is where you can add the values that you want to store in the list (in our case we will store the names of all the eight objects).

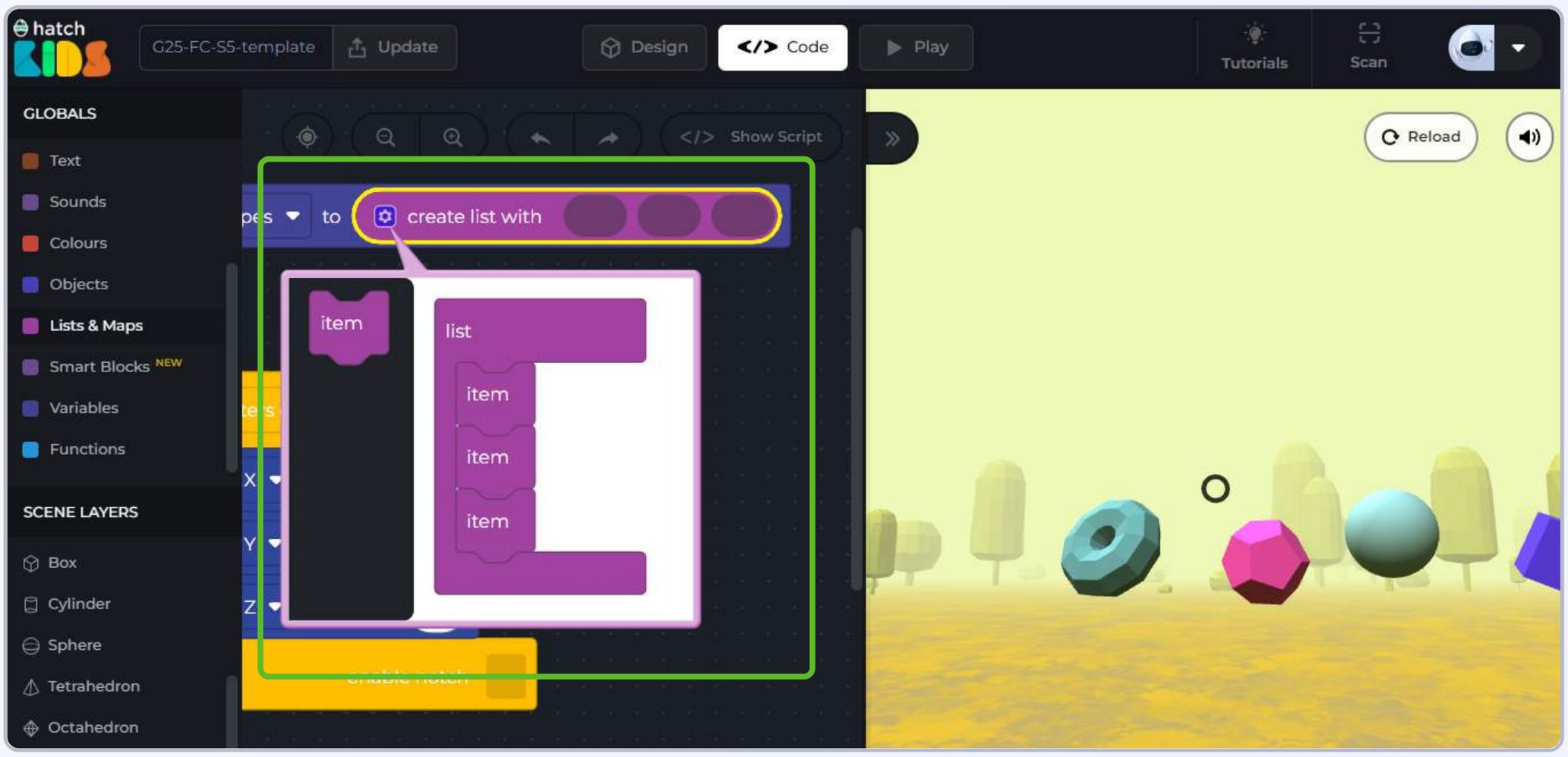


Since a list can store multiple values, you can modify the number of empty spaces available in the block

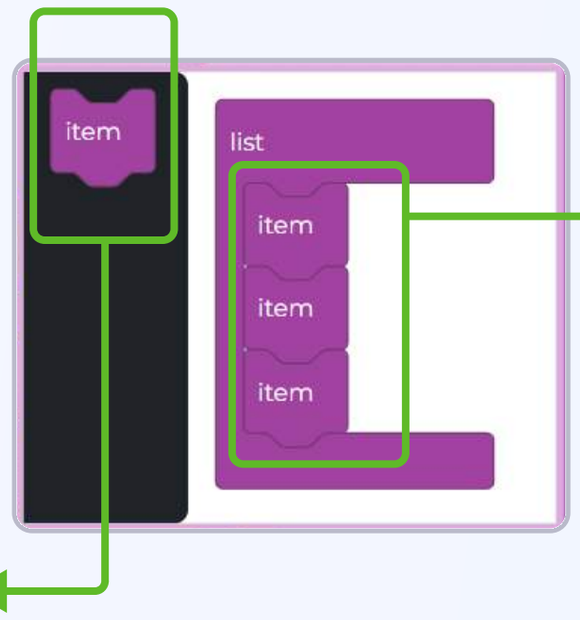
Click on this button to modify the number of empty spaces available in the list block

We want to add the names of the eight shapes in the list. So we need eight empty slots.

Step 6: Click on the “  ” button inside the **“create list with block”**

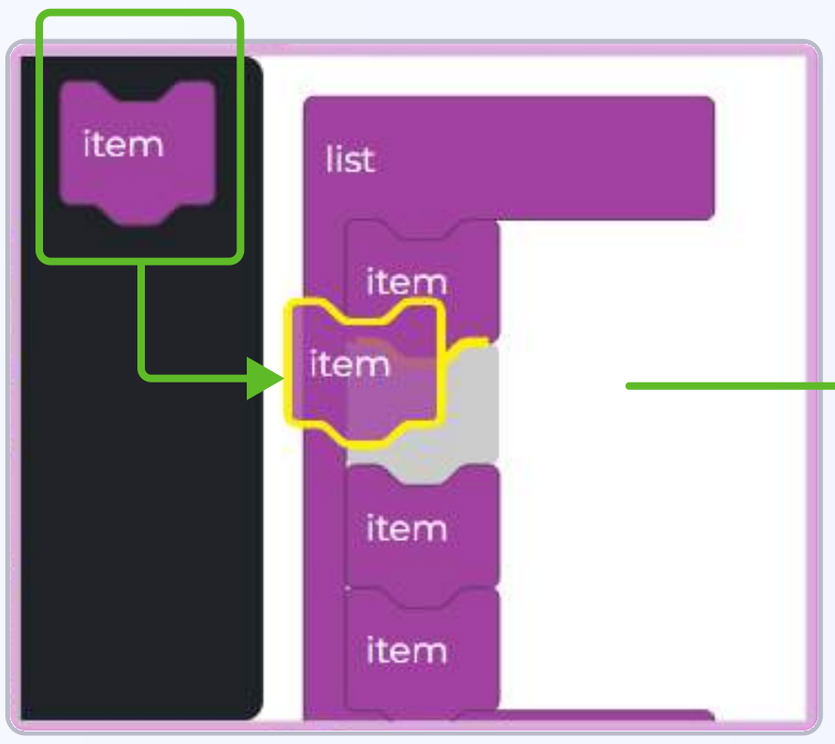


When you click on the  button, a pop up appears on your screen that looks like this.



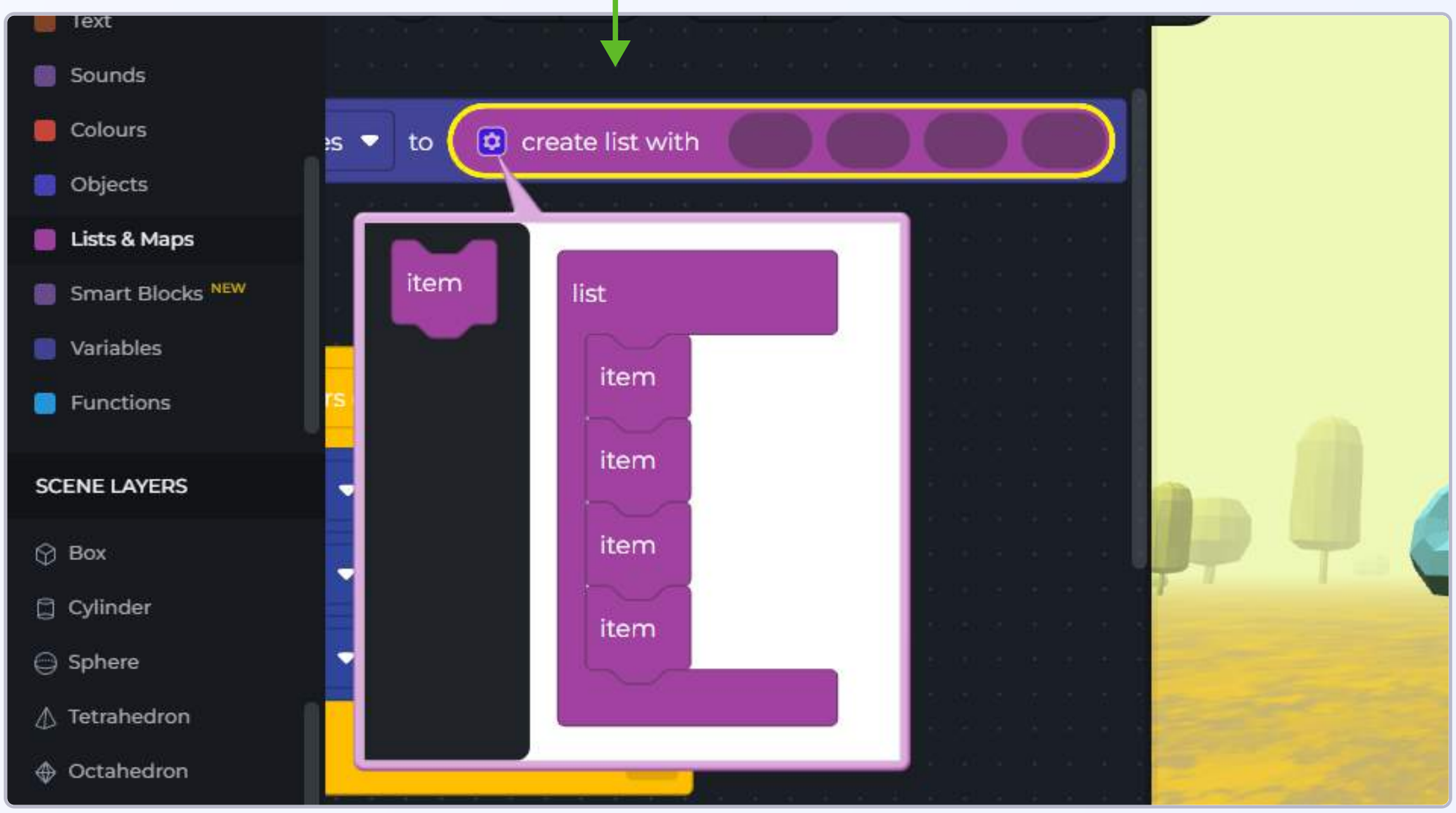
This space shows the number of items that can be added in the list, in the image as you can see it shows 3 items, and that's why you see the 3 empty spaces in the create list with block

To add more items (empty spaces) in the list, click on this block and drag it to the right side, and attach it with the other item blocks that you see



As you can see, the moment you add the new item block in the pop up box, a new empty space appears in the **"create list with"** block.

We need 8 empty spaces. so drag a few more item blocks in the white pop-up box.



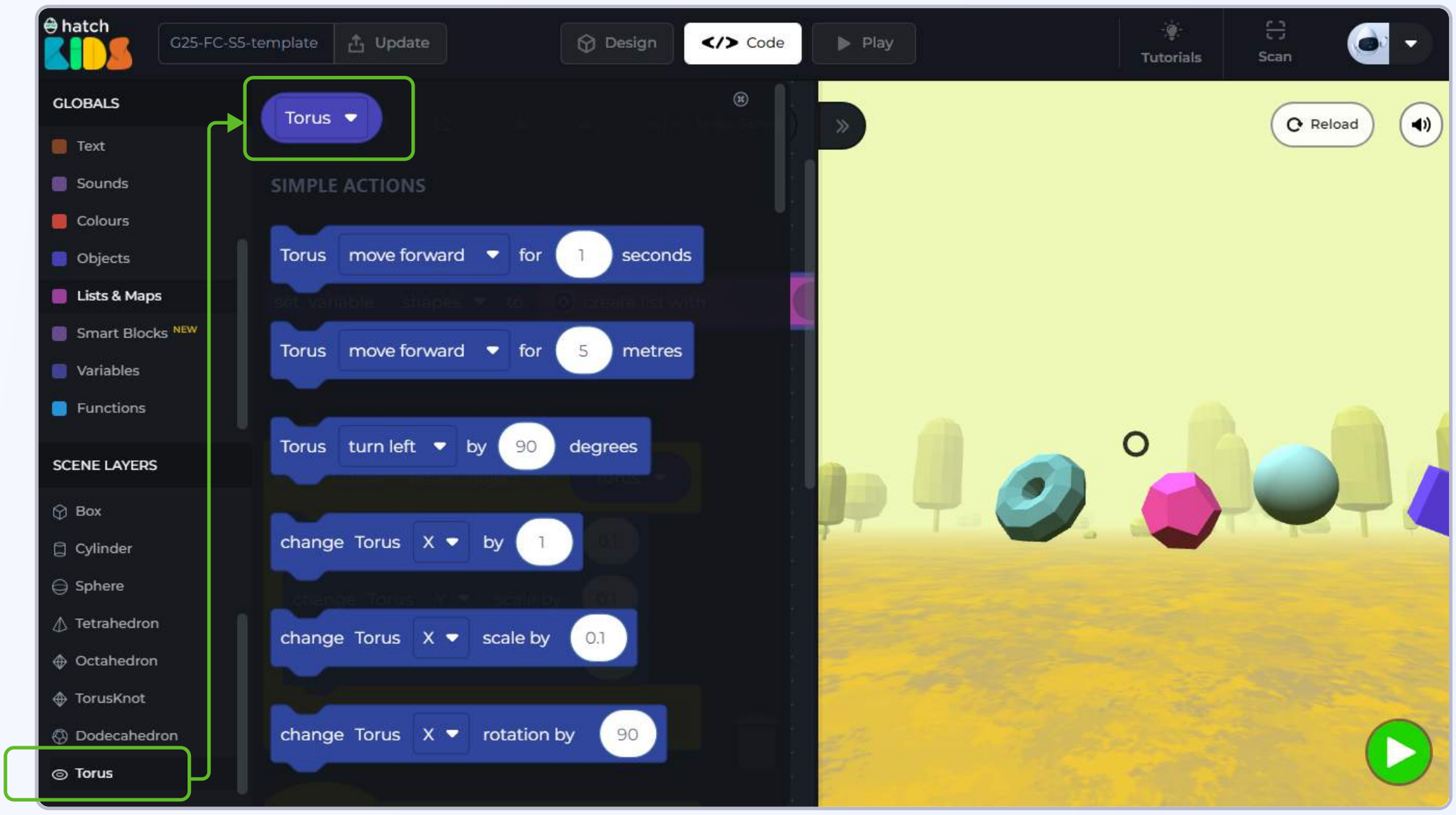
Step 7: Modify the **“create list with”** block to have **eight empty spaces.**

Step 8: Let’s fill those 8 empty spaces with the names of the objects in the project.

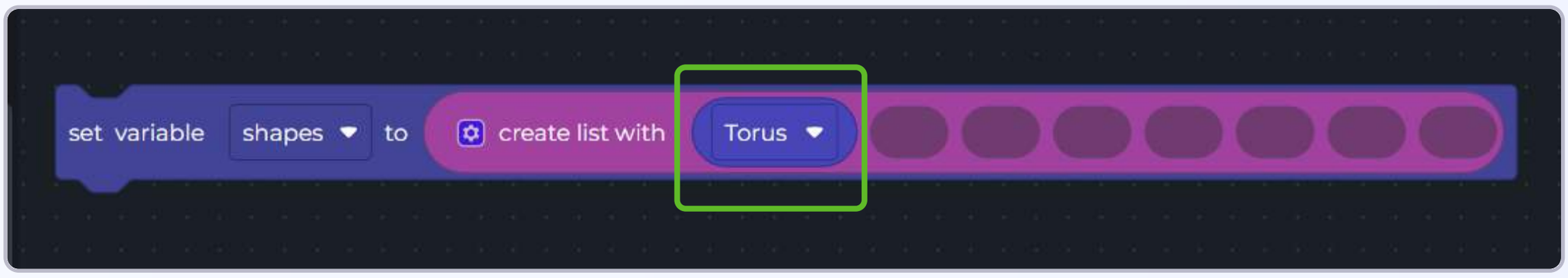
The order in which you add the names in a list is very important. Let’s add the names in the list based on the objects appearing from left to right.

First is **“Torus”.**

Click on the name **“Torus”** in the **left panel,** and **drag out the very first block** that just contains its name.



Step 9: Drag out the block that says **“Torus,”** and attach it in the first empty space inside the **“create list with” block.**

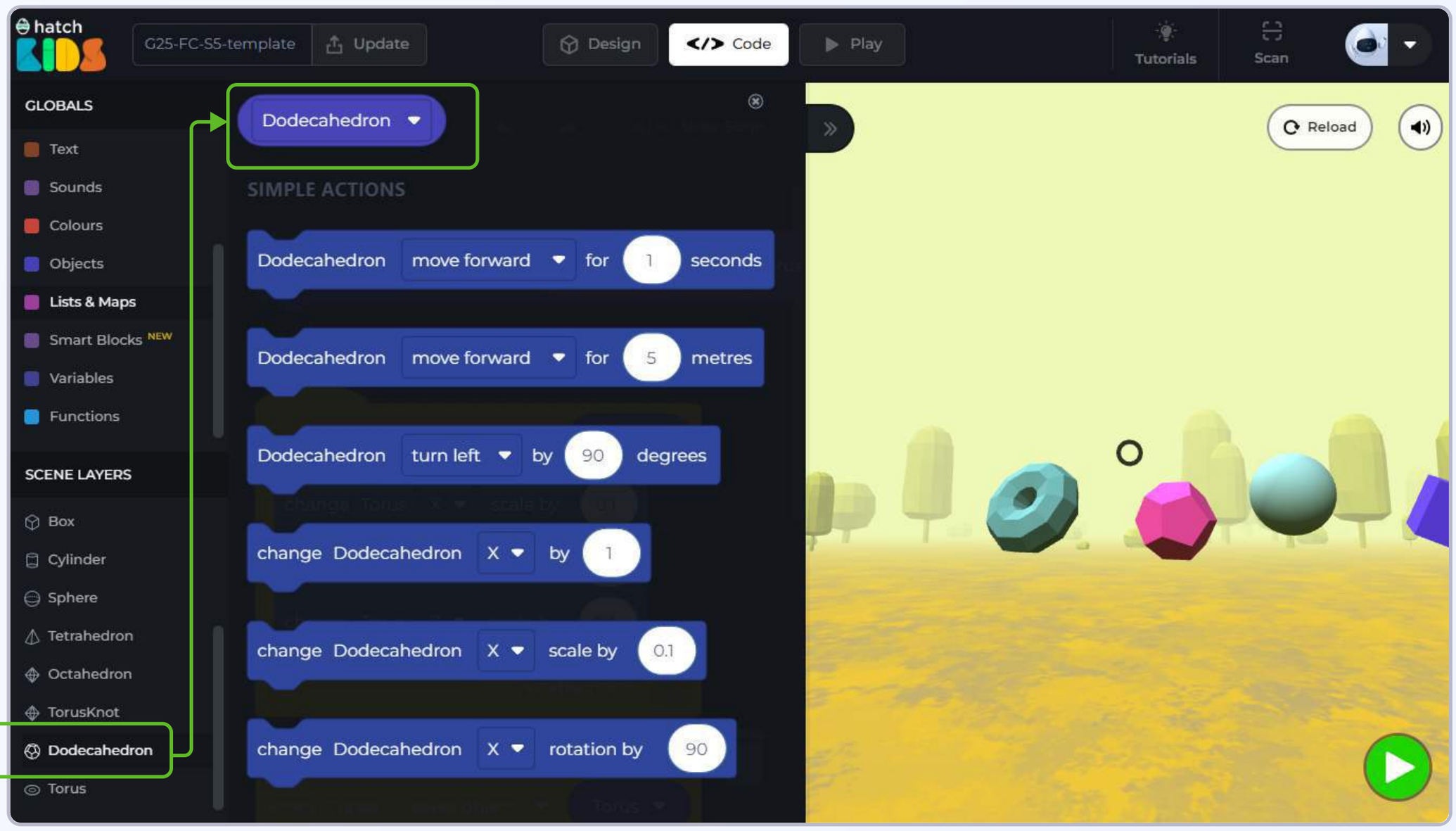


In the project, the next object on the right side of torus is **“Dodecahedron”.**

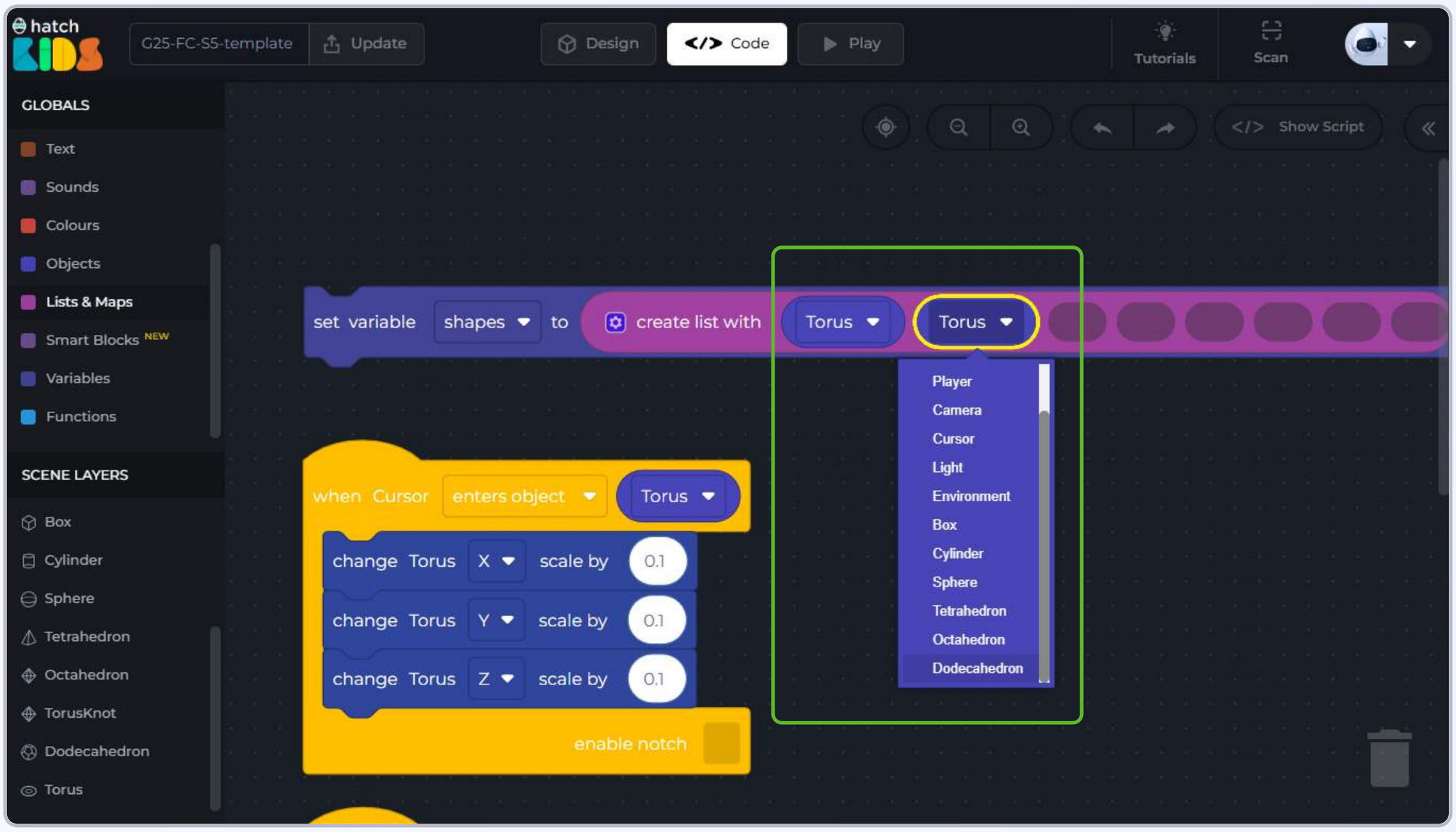
We need to attach the **“Dodecahedron”** block in the next empty space of the **“create list with” block.**

There are two ways to do this:

1. You can click on on the “Dodecahedron” name on the left panel and drag out the topmost block and attach it in the create list with block.



2. Or, you can duplicate the “Torus” block that is already inside the create list block, and then select the name dodecahedron from the drop down list.



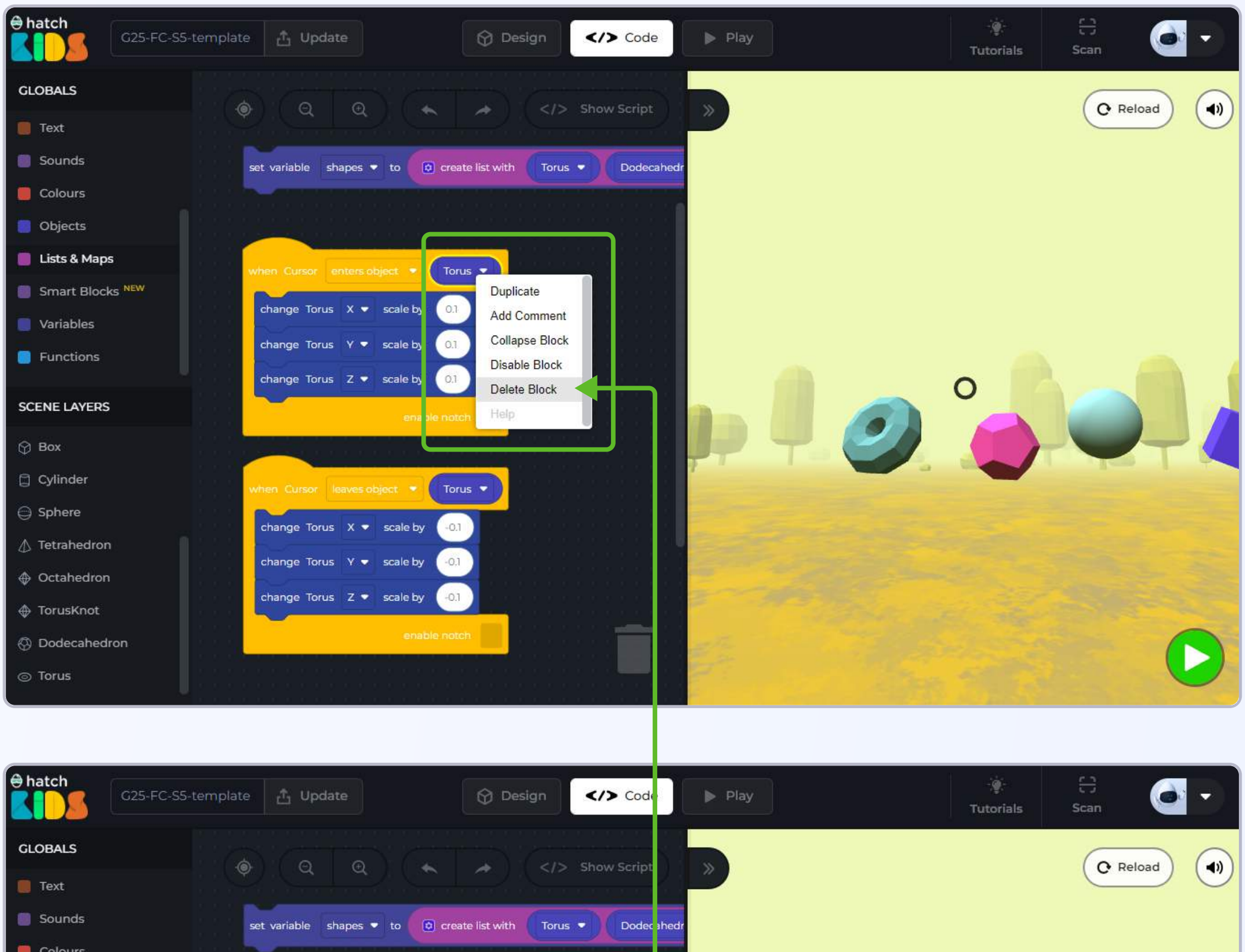
Step 10: Now that you know how to add the name blocks of all the shapes. Fill in all the empty spaces in the “create list with” block with the names of the objects in the order in which they appear from left to right. And that would be:

Torus, Dodecahedron, Sphere, Box, Tetrahedron, Cylinder, Octahedron, TorusKnot

So your final list should be as:

With the list ready, we can now modify the cursor entered object and cursor leaves object block, so that it works for all the 8 objects present in the game.

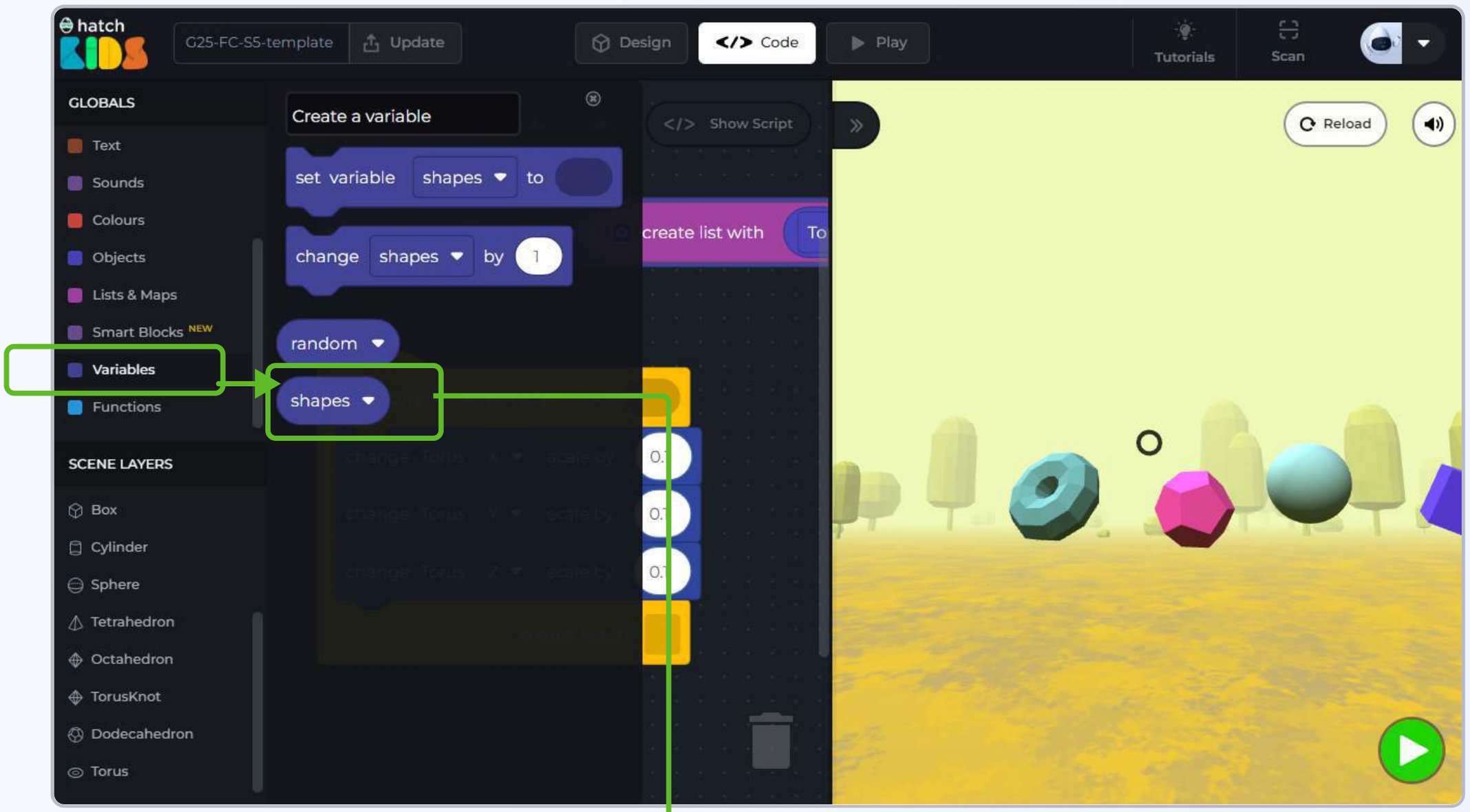
Step 11: In the **“when cursor enters object Torus”** block, right click on the block that says **“Torus”** and **delete the block.**



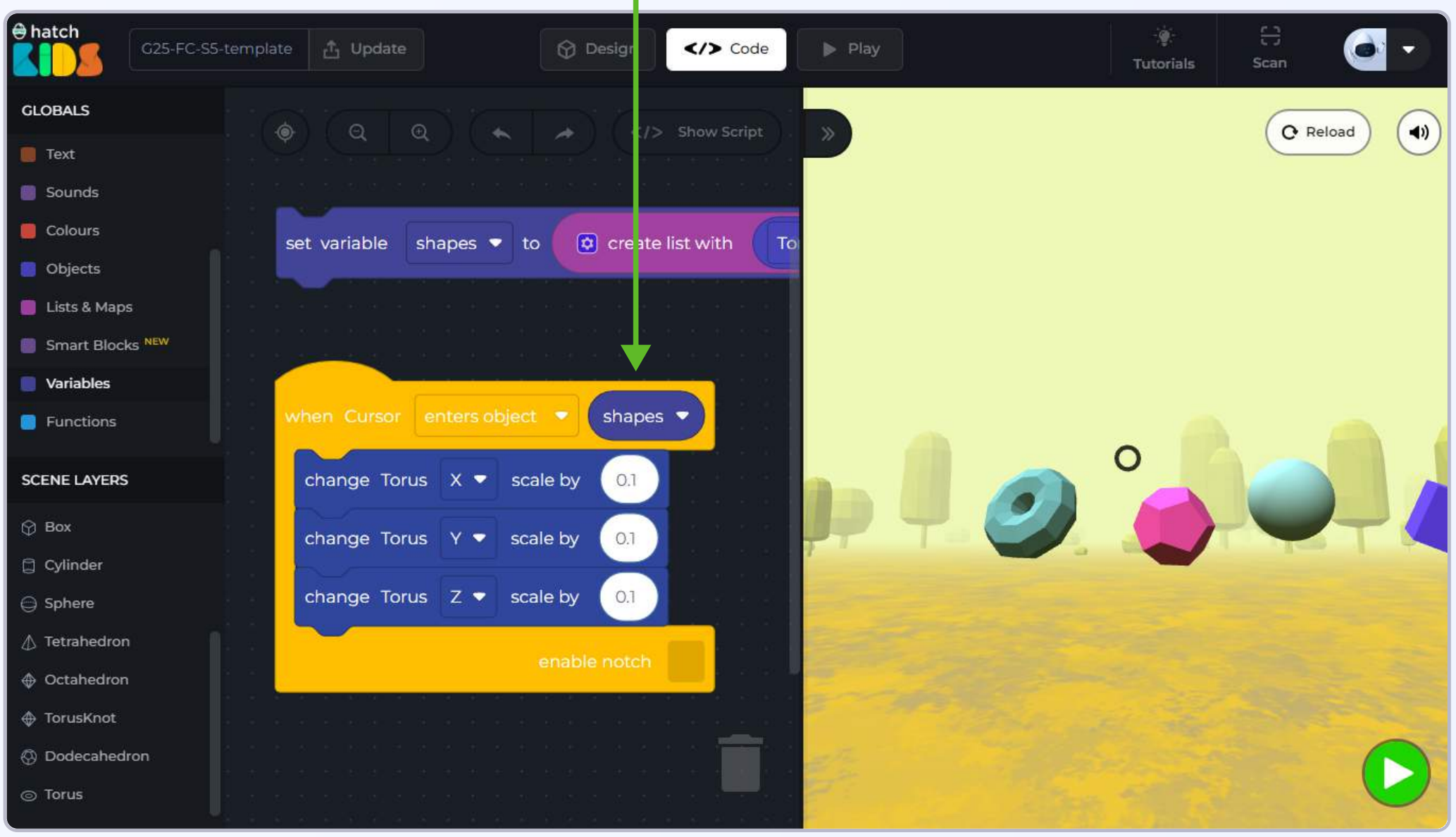
You can also **delete the entire “when cursor leaves object” block** as well.

Step 12: In the empty space of “when cursor enters object” block, we need to add the **variable name “shapes”** that contains our list.

Click on the **“variables”** section in the left panel, and drag out the block that contains the name of the variable **“shapes”**



Step 13: Attach the **“shapes”** block inside the blank space of the **“when cursor enters object”** block

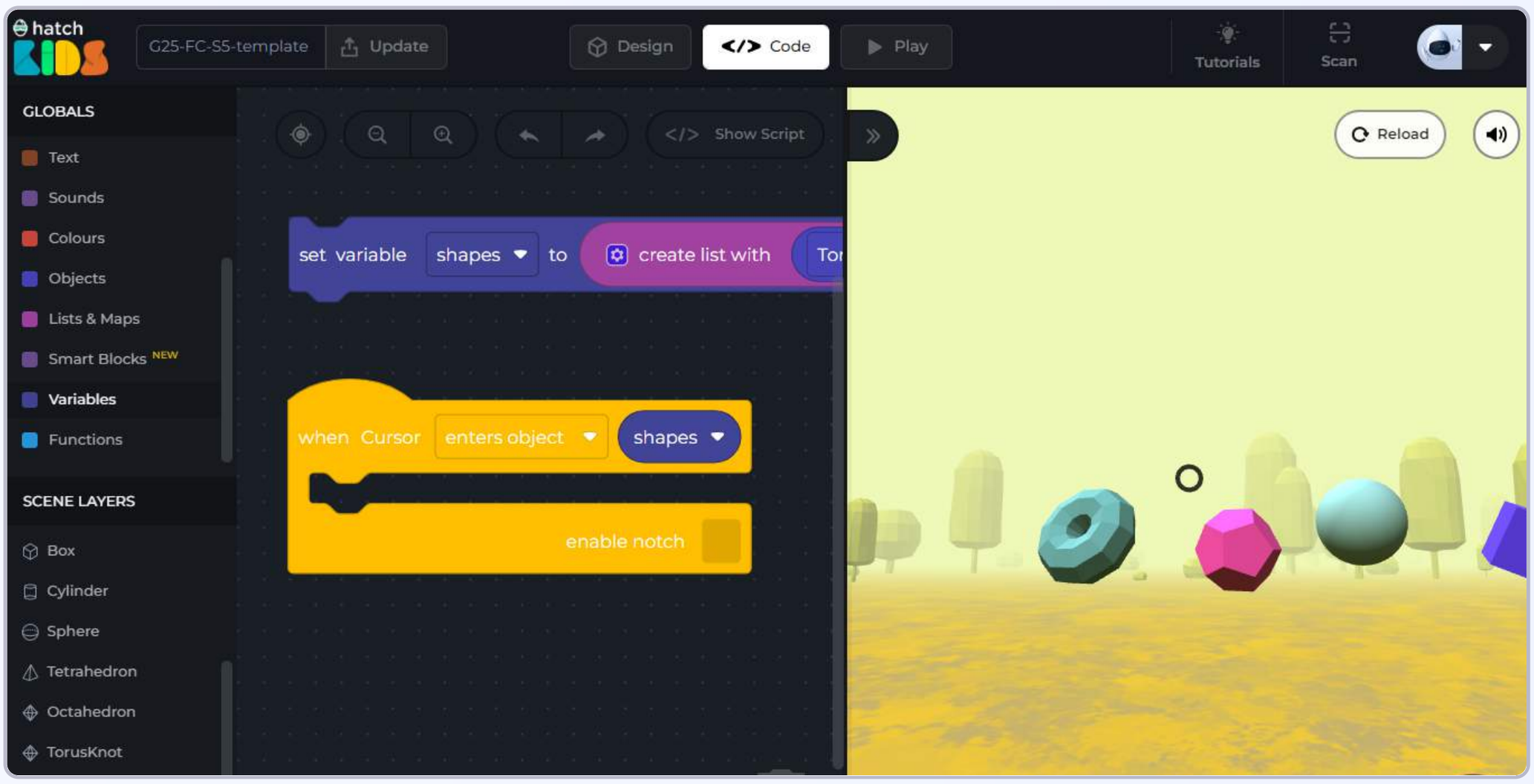


The variable “shapes” contains the list of all the names of the 8 objects present in our project.

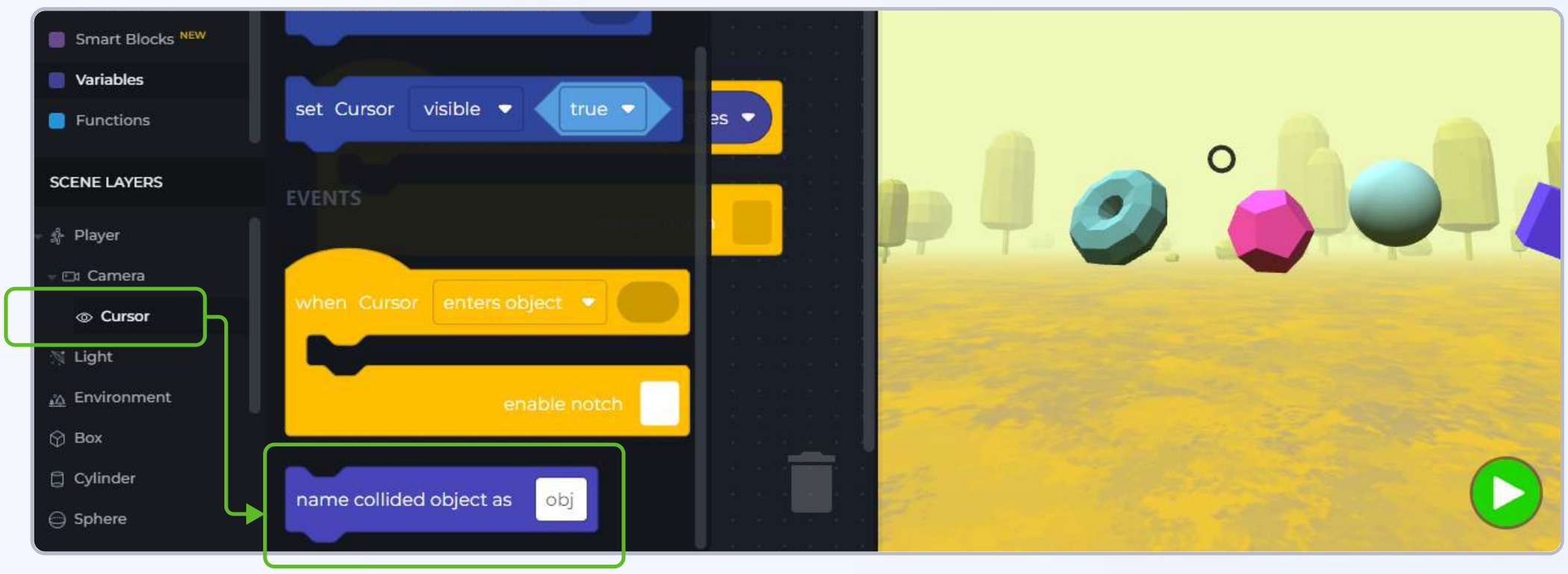
So when we place the **“shapes”** block inside the **“when cursor enters object”** block, we are telling our computer that when the cursor enters any one of the objects mentioned in the “shapes” list, then perform the activities mentioned inside the “cursor enters” block.

We need to implement our code such that, when cursor enters any one of the objects mentioned in the “shapes” list, then the size of that object increases.

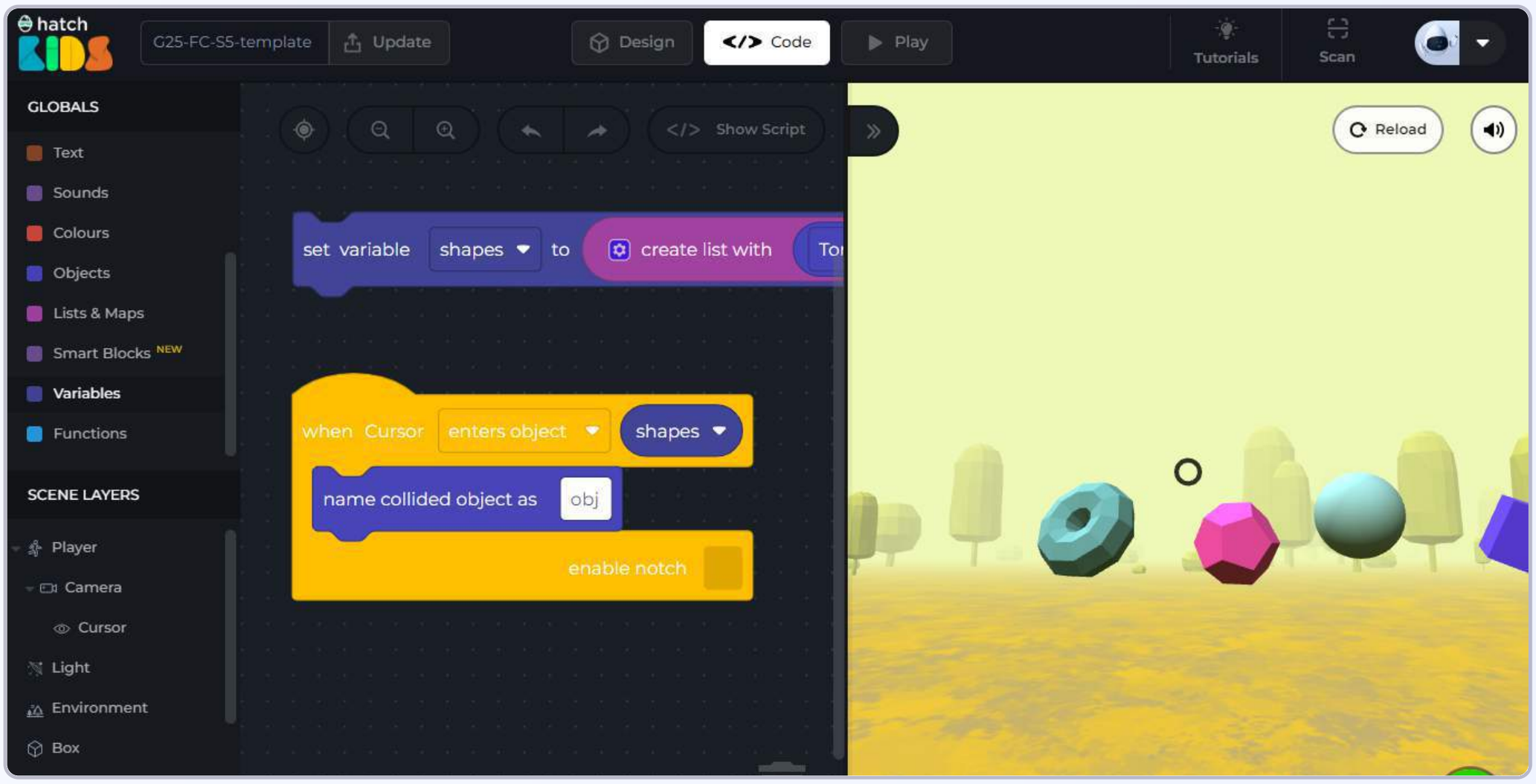
Step 14: Delete the **“change Torus X/Y/Z scale by 0.1”** blocks from inside the **“when cursor enters object shapes”** block.



Step 15: Click on the **“cursor”** option in the left panel. A list of blocks will appear. Go to the bottom of the list. You will see a block called, **“name collided object as”.**



Drag out the **“name collided object as”** block and attach it inside the **“when cursor enters object” block**



Let’s understand what is the use of the **“name collided object as”** block.

”Shapes” variable contains a list of names of eight objects present in the scene.
When cursor enters any one fo those objects, we want that specific object’s size to increase.

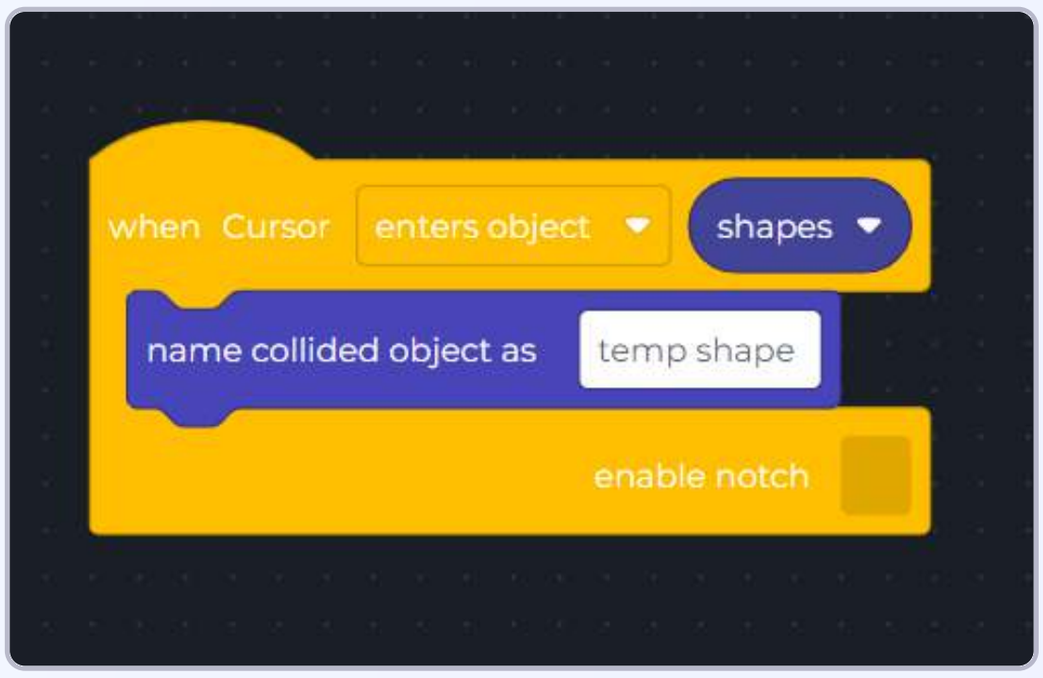
Let’s say the cursor has entered the “Sphere” object. Even though our computer will know over which object the cursor is moving, but we wont know that.

The **“name collided object as”** helps us provide a temporary name to the specific object over which the cursor is moving, so that we can perform actions like increasing its size, changing its position etc over that object by using the temporary name.

Let’s write the code and see how it works.

Step 16: The **“name collided object”** as block ends with a text input, where you can write the temporary name that want to give to the object with which the cursor is interacting.

In the input field of **“name collided object”** block, write **“temp shape”**

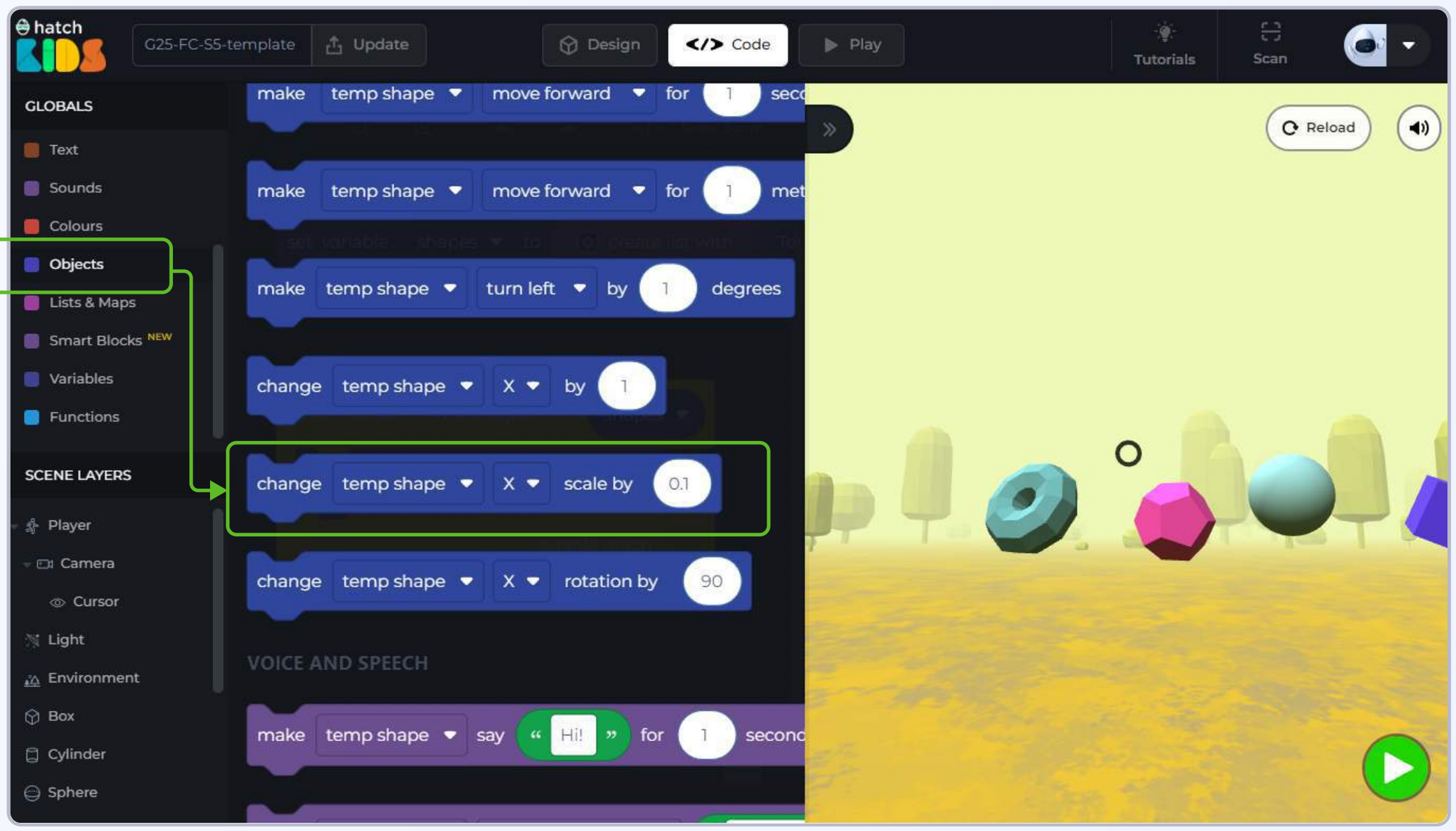


What this means is, when the cursor is moving over Sphere, the sphere object will be called by the name “temp shape”. And then when the cursor is moving over “Box”. then the box object will be called as “temp shape” by the computer.

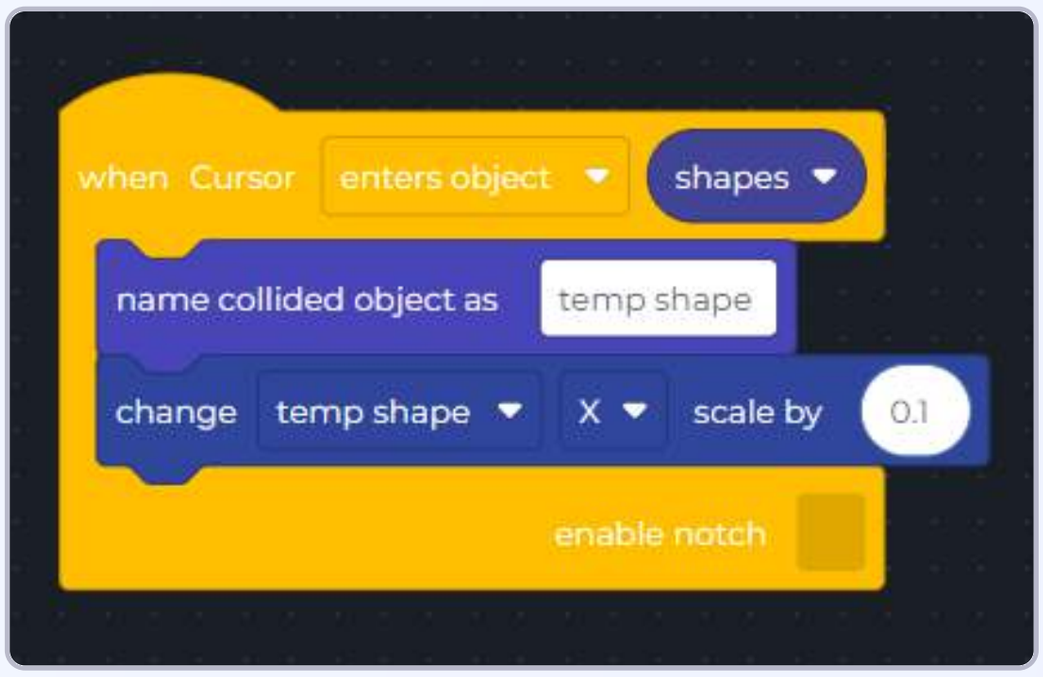
So now, if we want to increase the size of the object when cursor enters an object, we can just increase the size of “temp shape”.

Step 17: In the top half of the left panel on your workspace, there is a category of blocks called **“Objects”.**

Click on the **“Objects”** category in the left panel, and drag out the block that says **“change temp shape X scale by 0.1”**

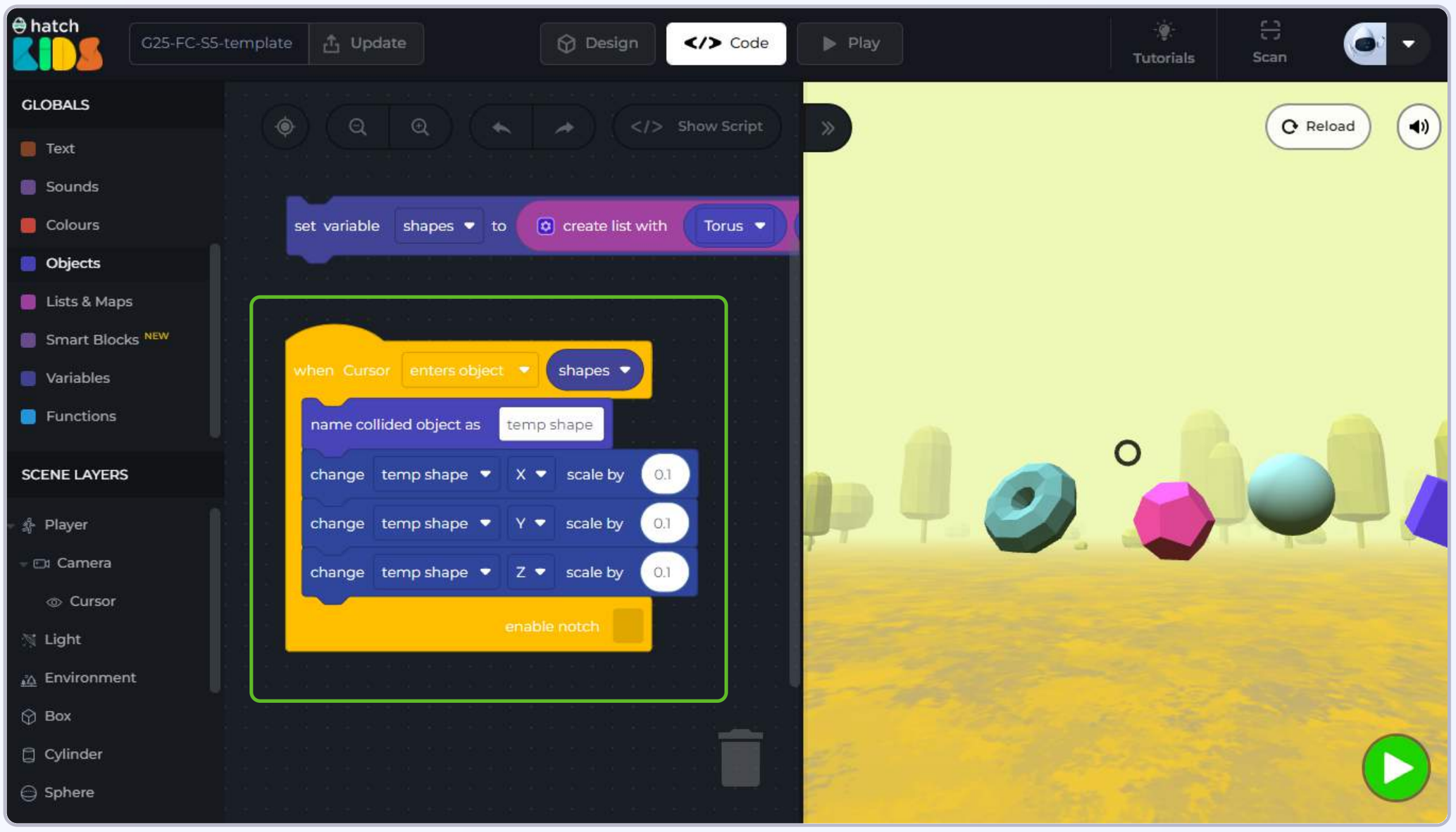


Step 18: Attach the **“change temp shape X scale by 0.1”** block inside the **“when cursor enters object shapes”** block as shown



Step 19: Just like at the beginning of this project, we added blocks to change the X, Y and Z scale of the Torus object,to see its size increase uniformly.

Similarly, we need to **duplicate the “change temp shape X scale by 0.1” block two times and the change the “X” to “Y” and “Z”** for each of them as shown



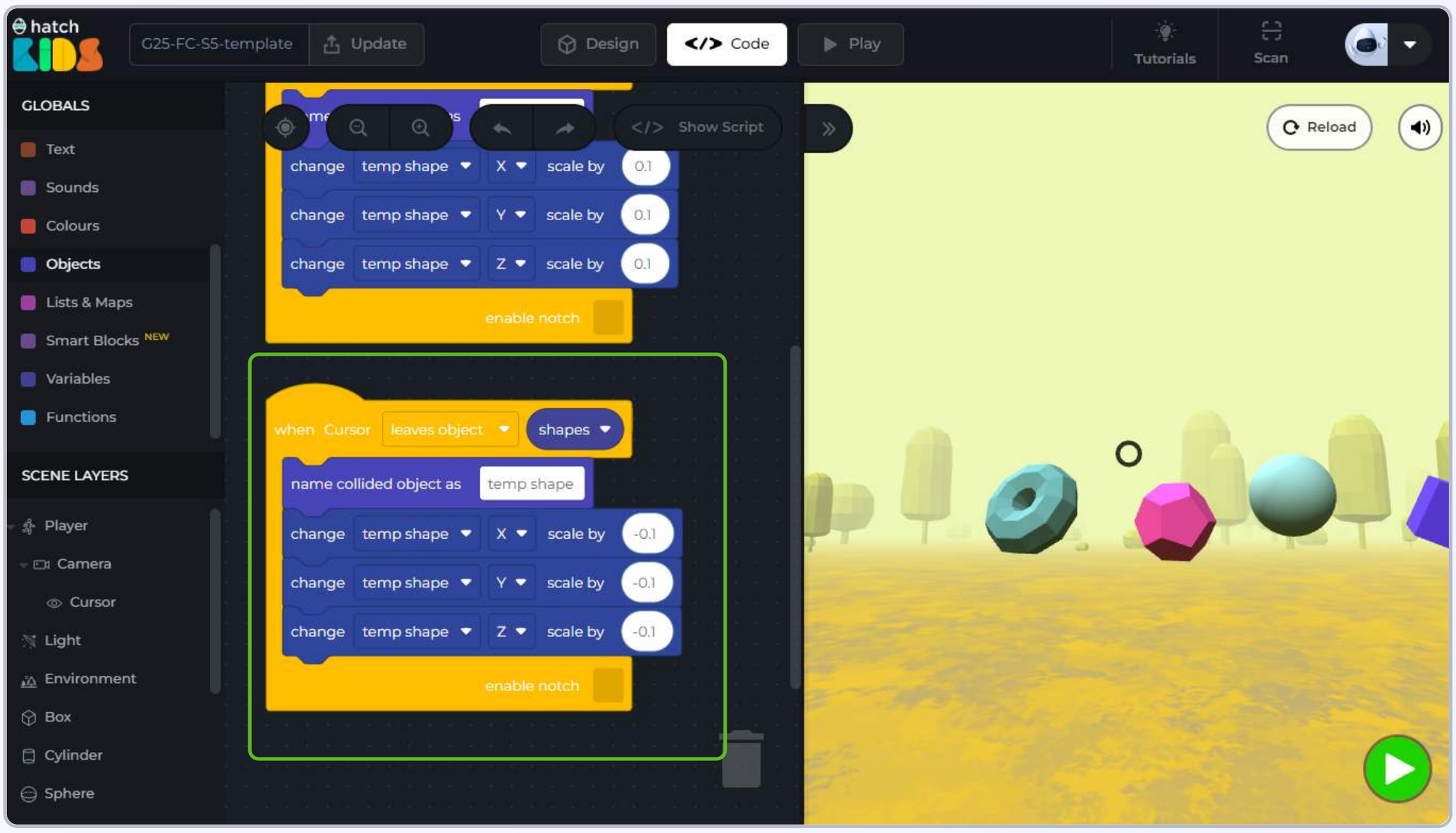
Reload your game and then click on the green play button to run the code.

Move your cursor around. You will notice that anytime the cursor enters any of the shapes, their size increases.

Step 20: Let’s code it so that when the cursor leaves any of the objects, their size decreases and goes back to normal.

just like we did at the beginning of this project:

1. Right click and duplicate the **“when cursor enters object shapes”** block
2. In the new copy, click on the **“enters object”** drop down and select the **“leaves object”** option
3. Since we want the scale to decrease when the cursor leaves any of the objects, so **change the number “0.1” to “-0.1” inside all the scale blocks.**



Reload your game and then click on the green play button to run the code.

Move your cursor around. You will notice that anytime the cursor enters any of the shapes, their size increases and when the cursor leaves the object, their size decreases.

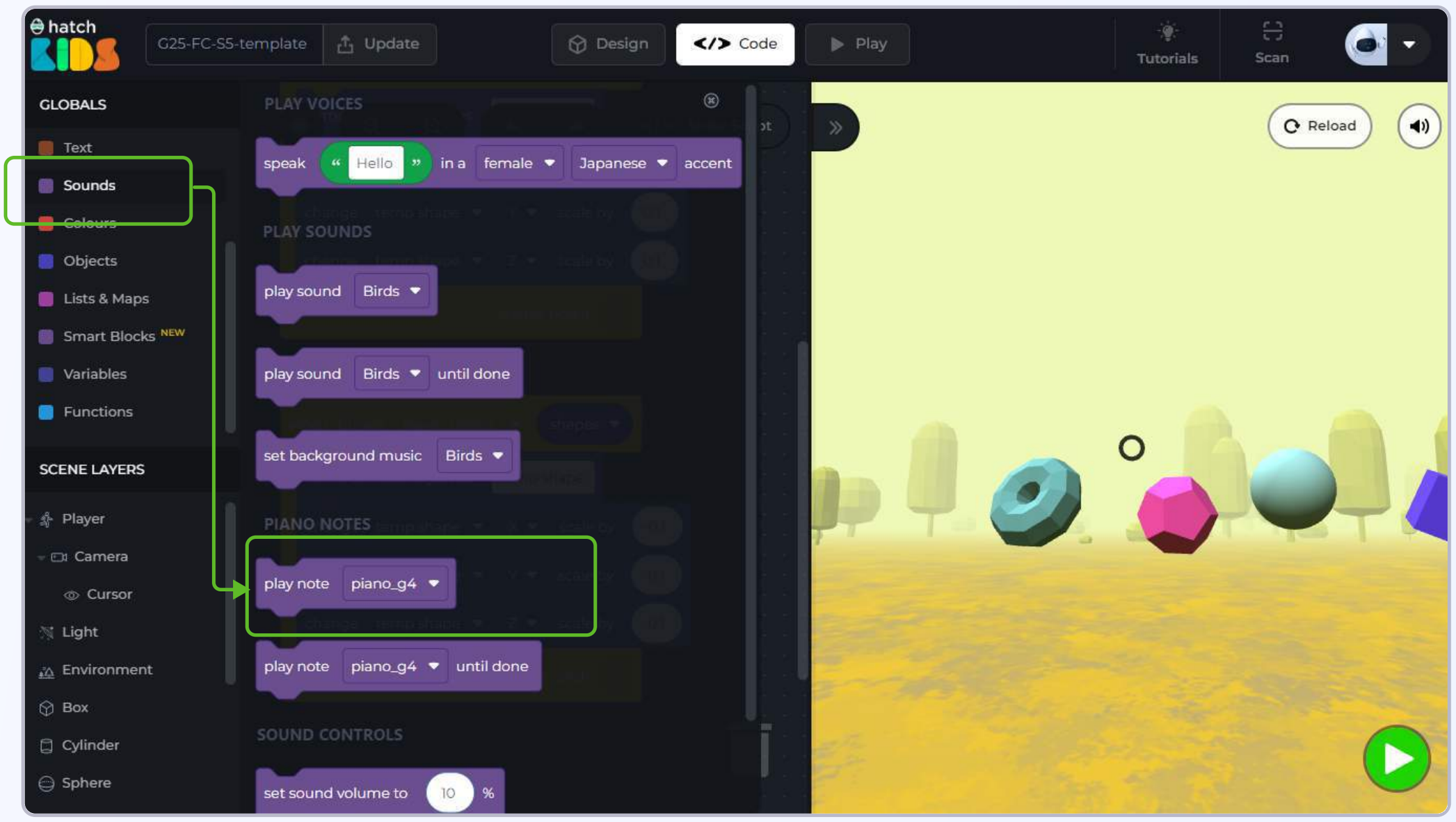
Let’s now add musical notes to be played when the cursor enters each of these objects.

Objective 4: Understanding if-else conditions

We have coded the basic interaction of the cursor with all the objects present in the game.

Since we are making a virtual piano, any time the cursor enters a specific object, some musical notes need to be played.

Step 1: Click on the **“sounds” category** on the top half of the left panel. You will see a block called **“play note piano_g4”**



Drag the **“play note piano_g4”** into the workspace and attach it inside the **“when cursor enters object” block**

Reload your scene and then click on the green play button to run your code. Move your cursor around and you will notice, that any time the cursor enters one of the objects, the size of the object increases, and a musical note is played every time.

The only issue here is that, all the objects are playing the same note.
If we want to make a proper musical instrument, then all the objects should be playing different notes.

Reload your scene.

Step 2: Click on the drop down option on the “play note piano_g4” block, and you will see a list of musical notes available. We have to implement the code such that every object plays one specific note.

Here is a proper list of the notes that should be played by specific object to make our project sound like a proper musical instrument:

Object	Musical Note
Torus	piano_c4
Dodecahedron	piano_d4
Sphere	piano_e4
Box	piano_f4
Tetrahedron	piano_g4
Cylinder	piano_a4
Octahedron	piano_b4
TorusKnot	piano_c5

To make different objects play different notes, we will have to use the concept of “conditions” in programming.

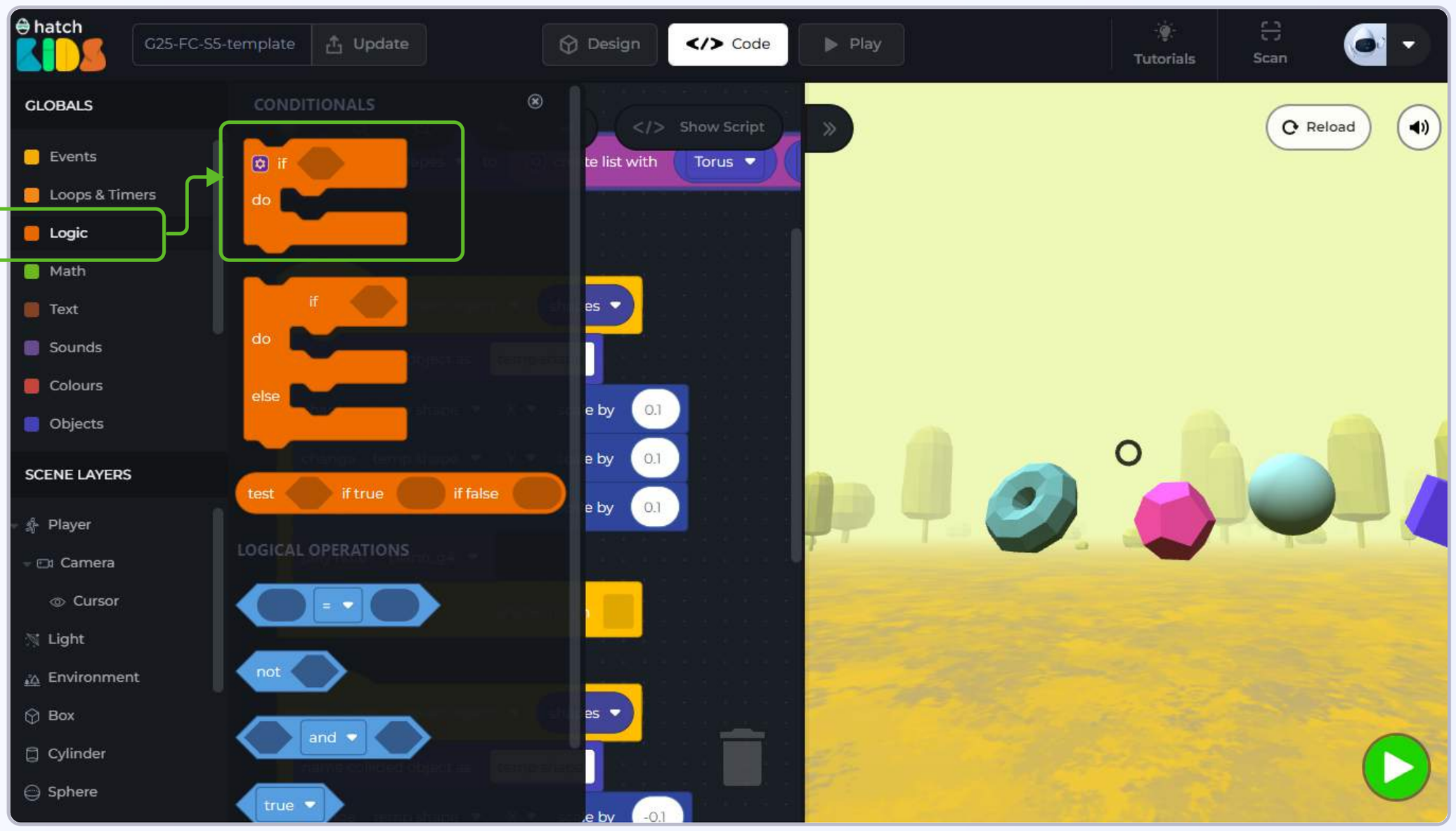
We know that at any point of time, the object over which the cursor is hovering has the name “temp shape”.

So we can write the code such that

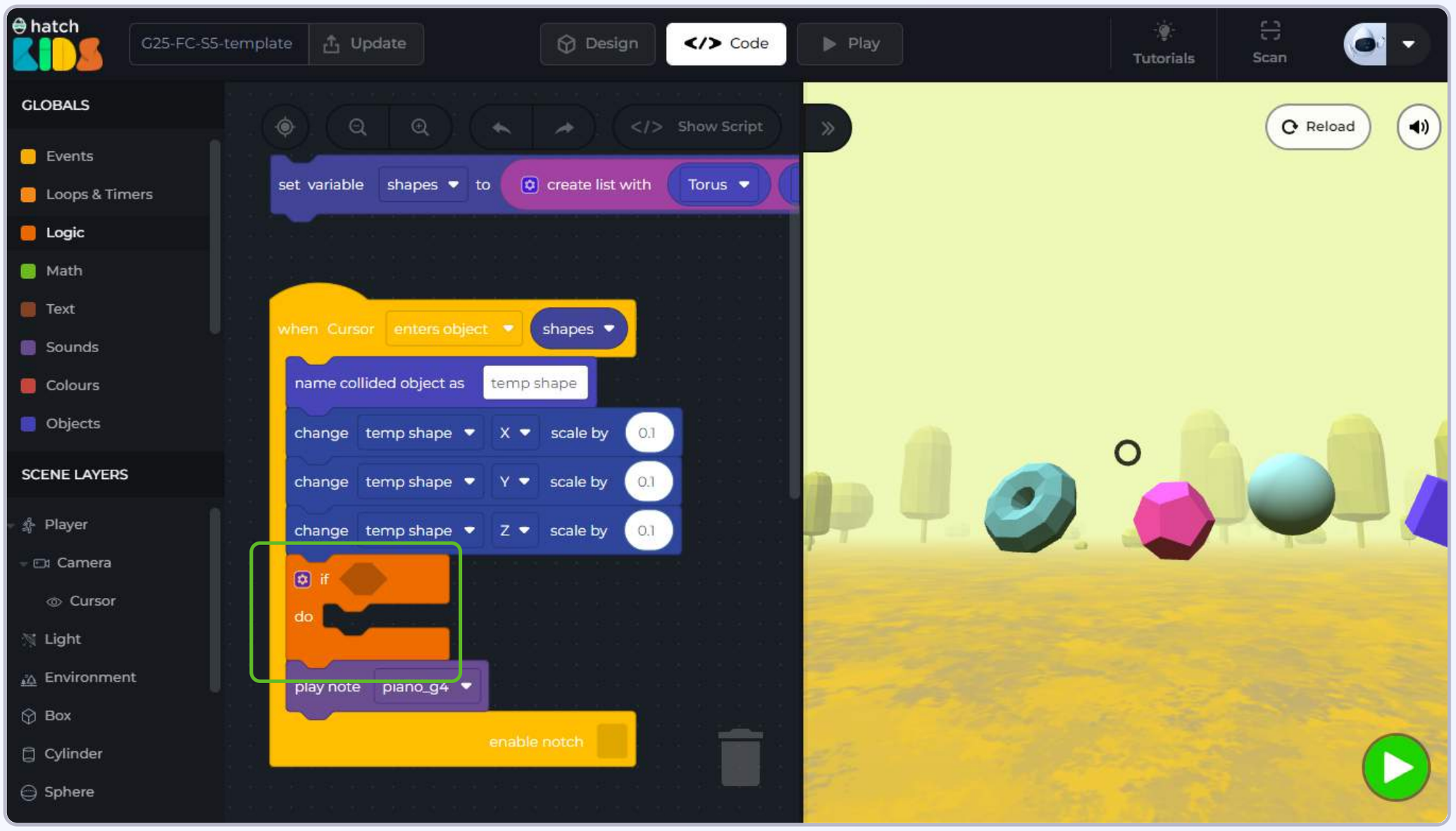
- if the **temp shape is torus** then **play the piano_c4** note
- if the **temp shape is dodecahedron** then **play the piano_d4** note
- if the t**emp shape is sphere** then **play the piano_e4** note …. and so on

These are called conditional statements in programming, Lets see how we can implement blocks with conditions

Step 3: Click on the **“Logic”** section in the top half of the left panel of the workspace.

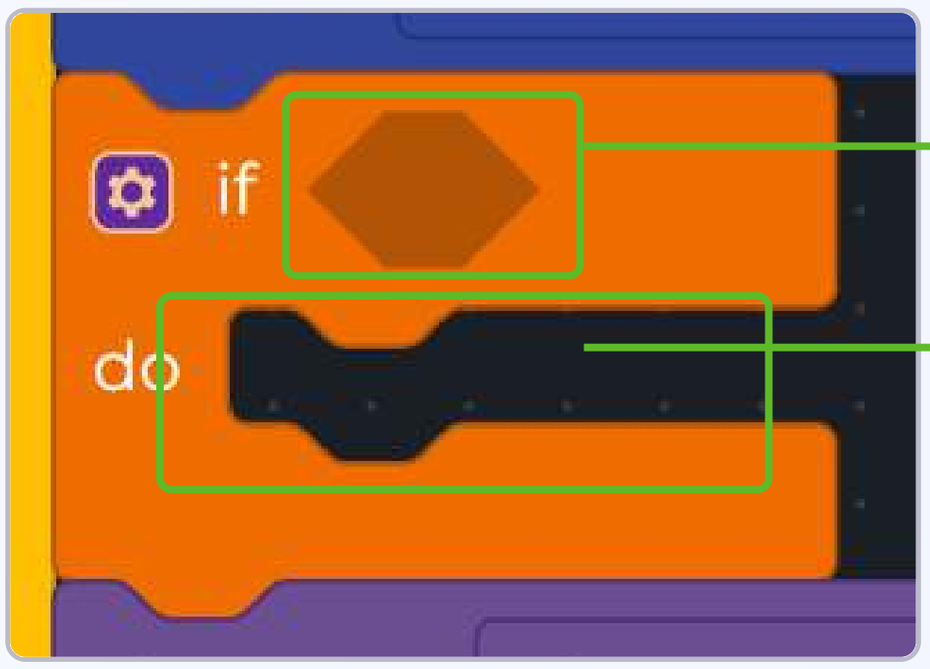


Drag out the very first block from the top that says, **“if ____ do _____”**, and attach it inside the **“when cursor enter”** block



The “if____ do ____” block is a conditional block.

Let’s understand its structure:



This is where you define the condition for the code to run

This is where you define the actions that need to happen when the condition that you have defined becomes true

(For eg: in our case -- if temp shape is torus then play piano_c4 note

if temp shape is torus ---- is the condition
play piano_c4 note -------- is the action

only when temp shape is torus, the the c4 note will play

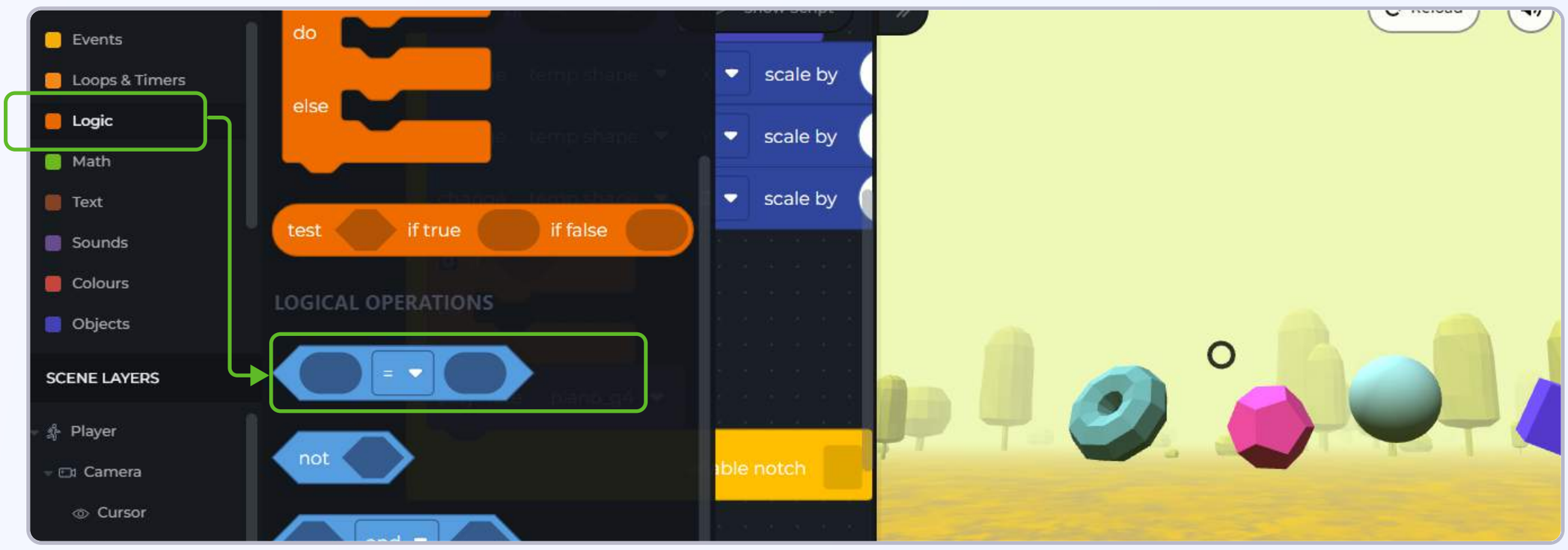
if the condition defined is not correct then computer will not run the action code)

Now that we know how conditional blocks work, lets put them to use.

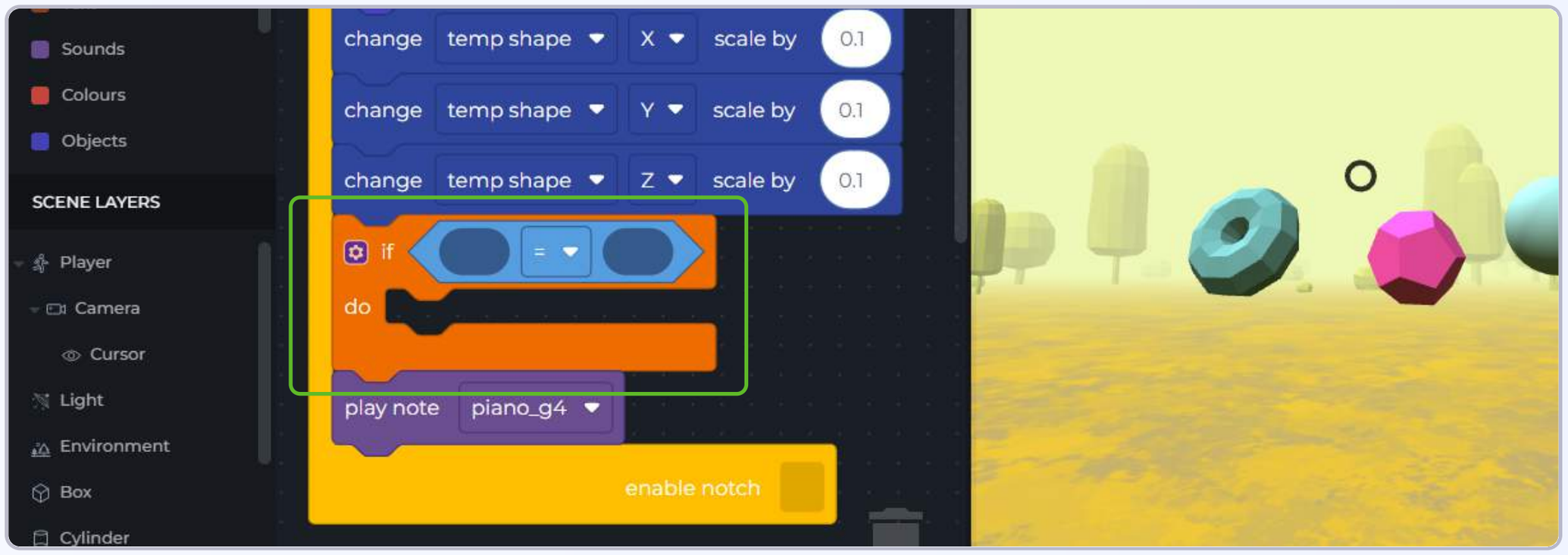
We can start by defining the first condition that -- if temp shape is Torus then play the piano_c4 note.

To code this:

Step 4: Click on the **“Logic”** section in the left panel and drag out the block that says 

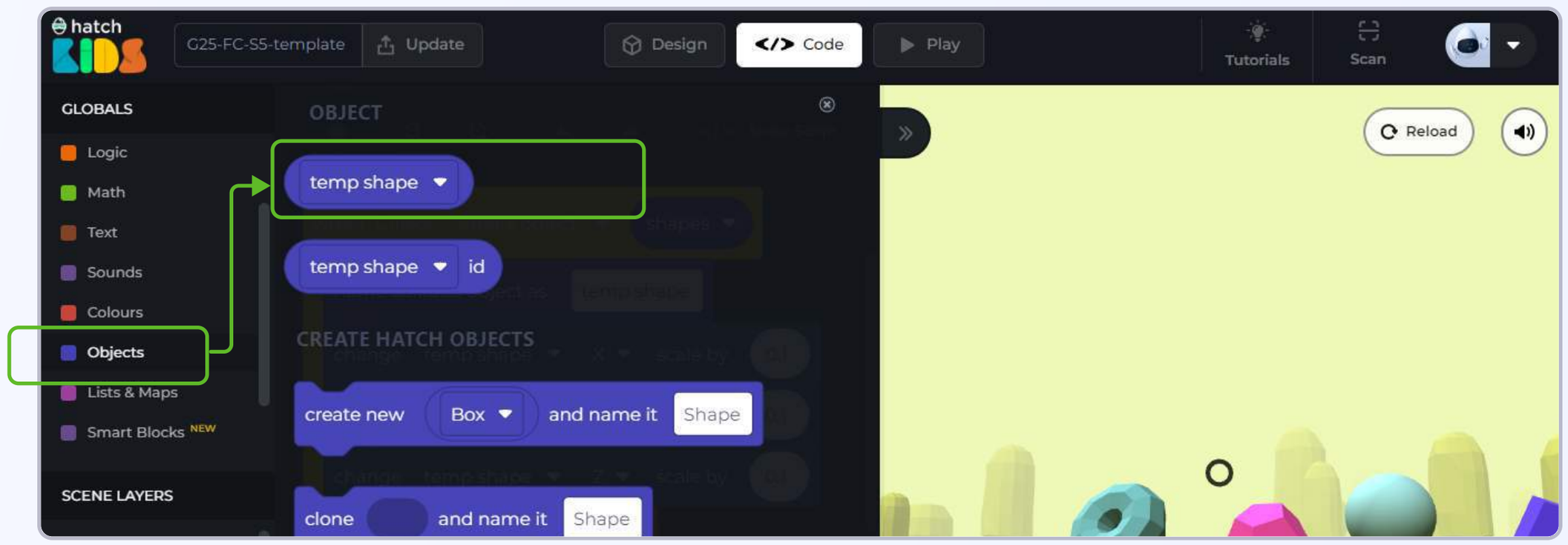


Step 5: Drag out the  block and attach it inside the **“conditions” space of the “if block”** as shown



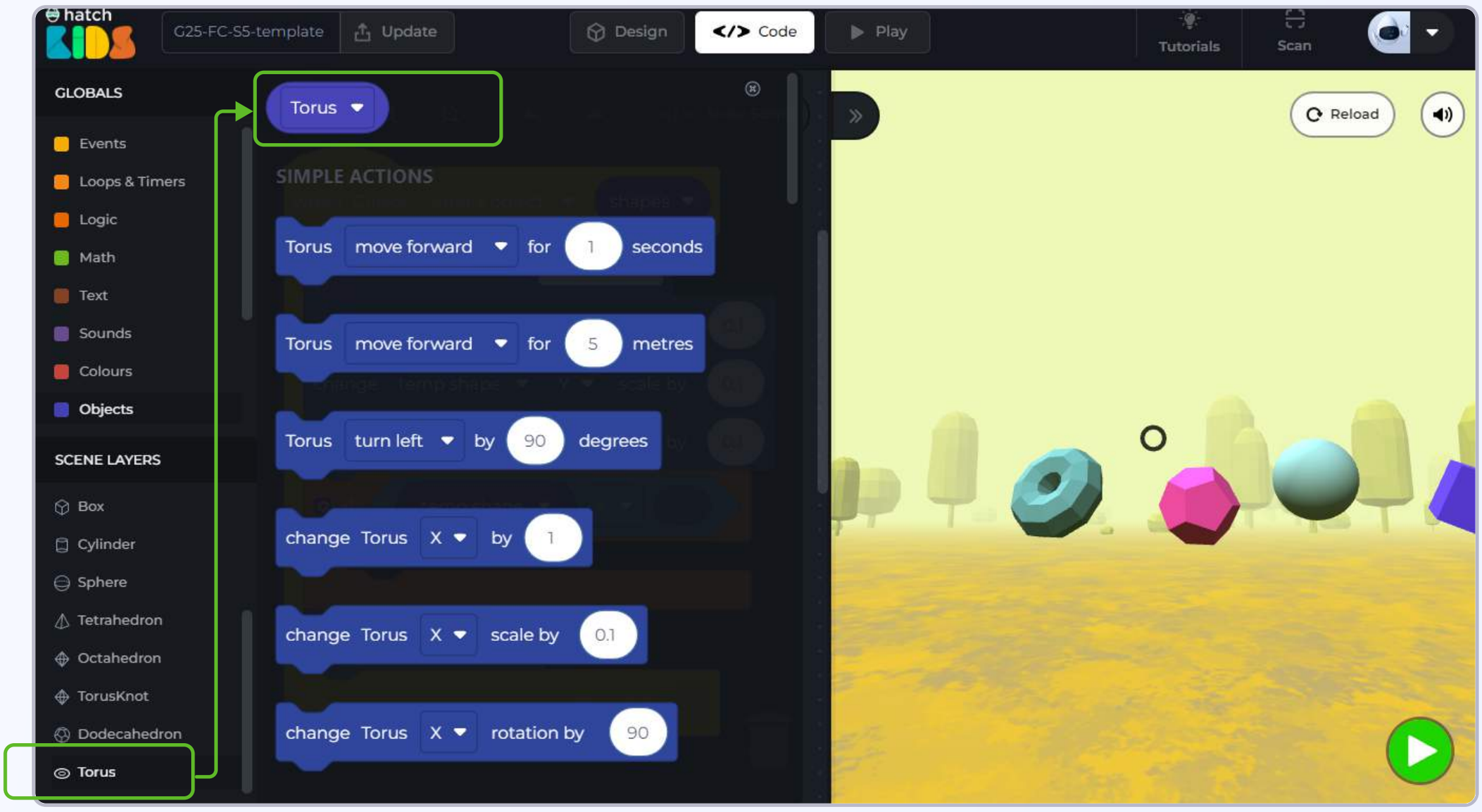
The code needs to say “if temp shape is Torus”, so we need to find individual blocks for “temp shape” [you can find this in the objects section in the top-left panel] and we need an individual block for the name “Torus”--[you can find this in the left panel by click on the torus name]

Step 6: Click on the **“Objects”** section in the top left panel, and you will see a block that says **“temp shape”** at the top.

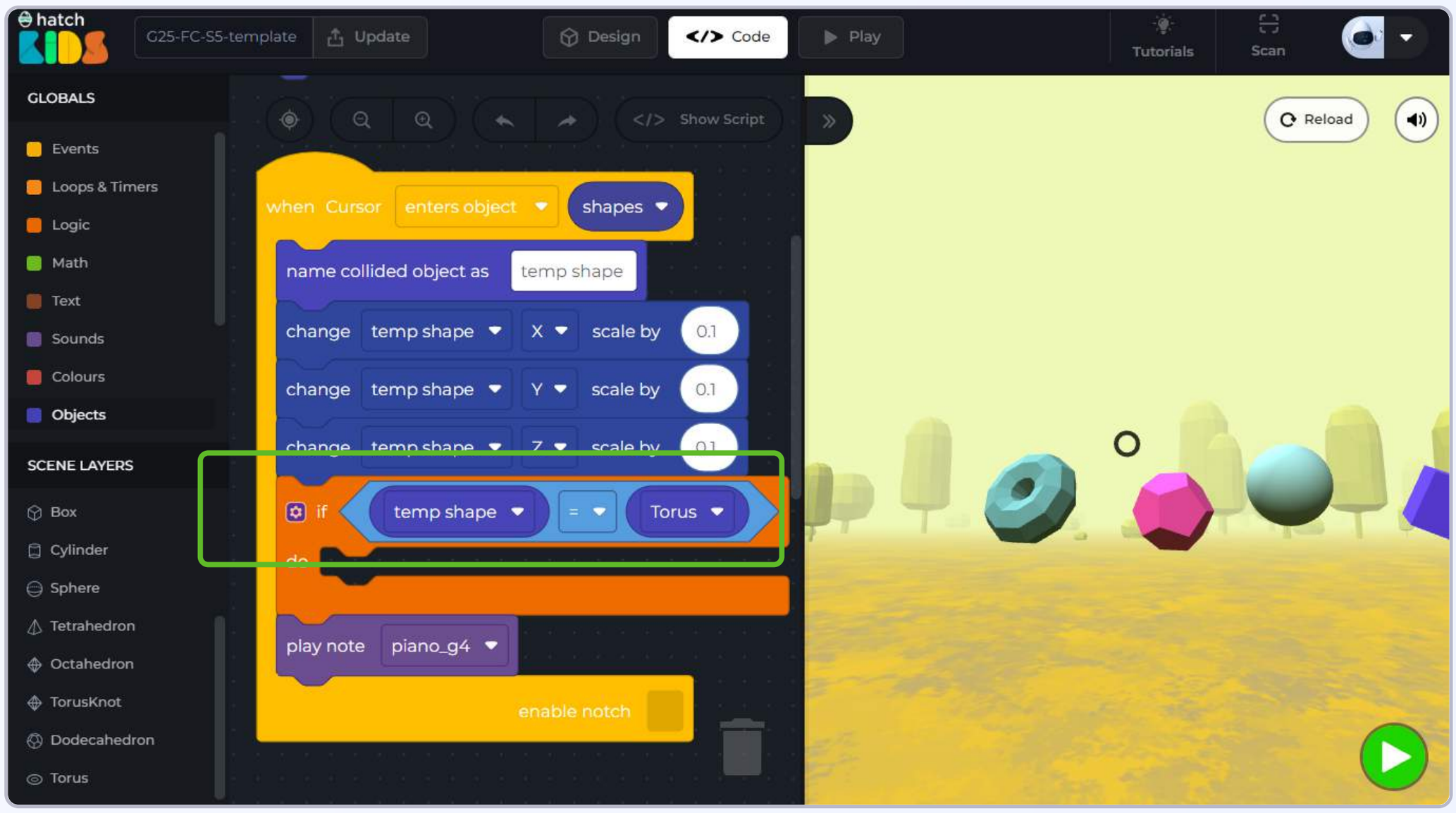


Drag this block into the workspace and attach it as shown

Step 7: Click on the name **“Torus”** and drag out the very first block from its list that just says its name.



Drag the **“Torus”** block into the workspace and **attach it inside the condition block** as shown.



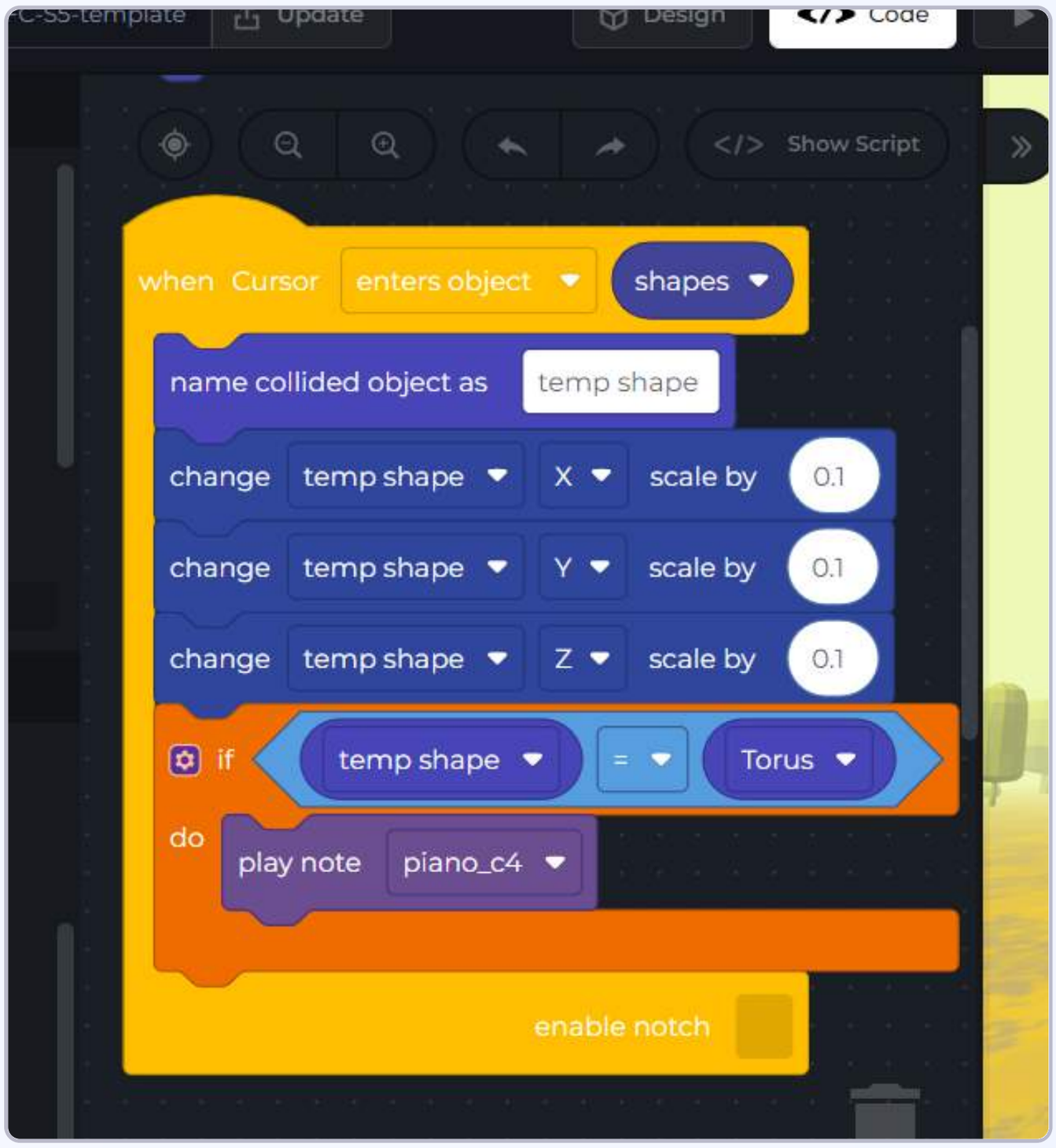
The if-condition block in the workspace now says “if temp shape = Torus do _____”

We need to define what action should happen if the cursor enters the Torus shape, that is, if the temp shape = Torus.

As per the musical note table, the Torus object should play “plaino_c4” note.

Step 8: Drag out the “play note” block form below the if block, and attach it inside the if block as shown. Also from the drop down option of various piano notes, select the “piano_c4” option.

Your block should now read:

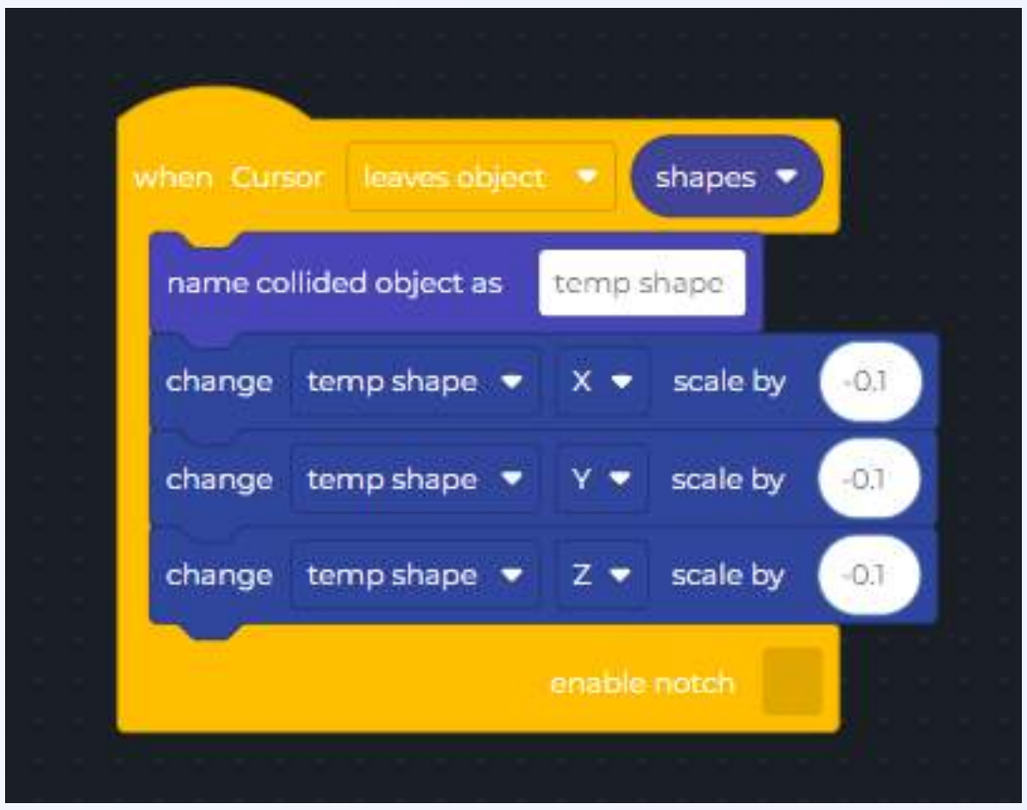


Reload your scene, and click on the green play button to run the code.

Now you will notice, as you move the cursor, the size of all the shapes are increasing and decreasing, but music is not being played by all the shapes like earlier. Now only when you move the cursor over the torus object, then you are able to here the music play..

Just like we added one if condition for Torus to play the c4 note, we can duplicate the if-condition block, and change the torus option to the names of the other objects and change the music notes as per mentioned in the table a few pages back.

Once completed your code would look something along the lines of:



Reload your scene, and click on the green play button to run the code.

Now you will notice, as you move the cursor, the size of all the shapes are increasing and decreasing, and every shape is playing a different musical note just like a musical instrument.

This completes your virtual piano project. You can login to your account, and give your project a name, and publish and share among your friends and you can even try building similar projects replicating other musical instrucments as well. Imagination is the limit here.

🎉🎉🎉 **END OF GUIDE** 🎉🎉🎉