# CE-324

# Homework-4

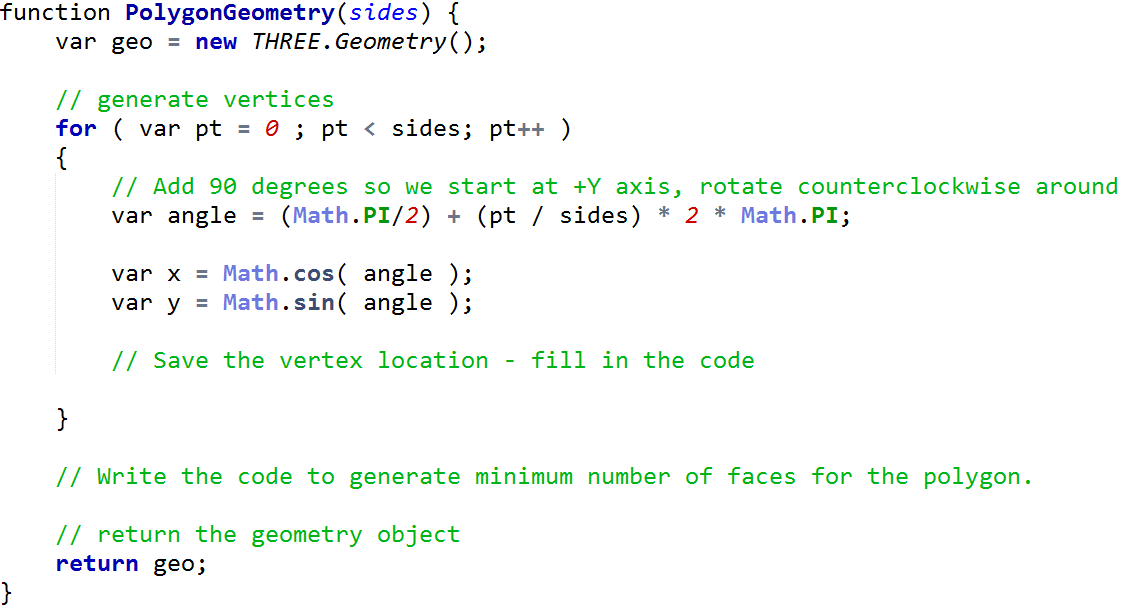
# Assignment-1: Make a Polygon Creator

งานของนักศึกษาในข้อนี้ คือ การเขียนโค้ด JavaScript ที่ใช้ library Three.js เพื่อ**สร้าง Polygon ออกมา โดยที่ Polygon** คือ วัตถุที่มีมุมเท่ากันและด้านเท่ากันหมด คำอธิบายของ assignment นี้ ให้ดูจากวีดีโอข้างล่าง

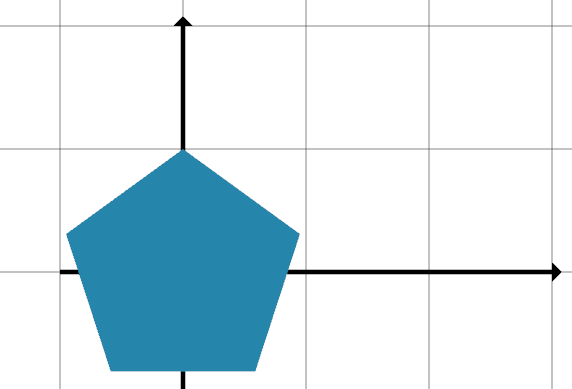
<https://www.youtube.com/v/V67w9xDbD58>

**สิ่งที่นักศึกษาต้องทำ**

1. เขียน JavaScript ฟังก์ชัน ชื่อ **PolygonGeometry()** ซึ่งรับ parameter เป็นจำนวนด้านของ polygon
2. เมื่อรับค่าจำนวนด้านมาแล้ว ให้นักศึกษาสร้าง polygon โดยใช้จำนวน triangles ที่น้อยที่สุด
3. Template ของฟังก์ชัน **PolygonGeometry()** แสดงดังข้างล่าง โดยอยู่ในไฟล์ “***polygongeometry1.html***” นักศึกษาต้องเพิ่มโค้ดใต้คอมเมนต์ YOUR CODE HERE กำหนดให้รัศมีของ polygon = 1 สำหรับงานนี้



**ตัวอย่างงานที่เสร็จแล้ว**



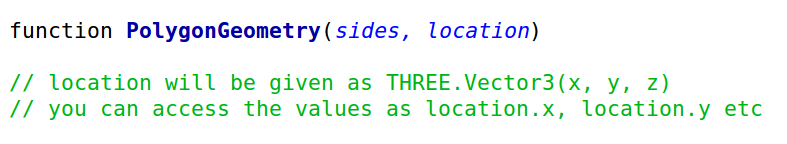
# Assignment-2: Move a Polygon to a Location

สำหรับ Assignment-2 เป็นงานที่ทำต่อเนื่องมาจาก Assignment-1 โดยให้นักศึกษาเขียนโค้ด JavaScript ที่ใช้ library Three.js เพื่อสร้าง **Polygon ที่มีจุดศูนย์กลางของ polygon อยู่ที่พิกัด (x, y, 0)** เมื่อ x และ y เป็นค่าที่ผู้ใช้กำหนดมา คำอธิบายเพิ่มเติมของ assignment นี้ ให้ดูจากวีดีโอข้างล่าง

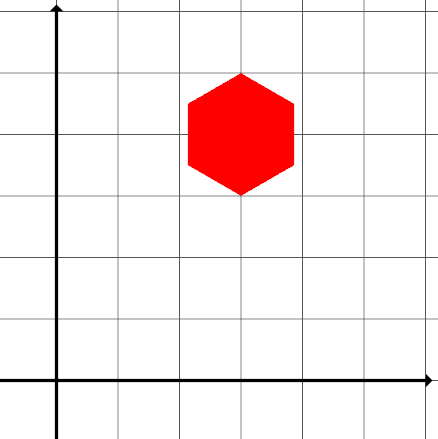
<https://www.youtube.com/v/Yaz1IHypGO8>

**สิ่งที่นักศึกษาต้องทำ**

1. เขียน JavaScript ฟังก์ชัน ชื่อ **PolygonGeometry()** ซึ่งรับ parameter เป็นจำนวนด้านของ polygon (sides) และ พิกัดจุดศูนย์กลางของ polygon (location)
2. เมื่อรับค่าจำนวนด้าน และ พิกัดจุดศูนย์กลางของ polygon มาแล้ว ให้นักศึกษาสร้าง polygon โดยใช้จำนวน triangles ที่น้อยที่สุด
3. Template ของฟังก์ชัน **PolygonGeometry()** อยู่ในไฟล์ “***polygongeometry2.html***” นักศึกษาต้องเพิ่มโค้ดในฟังก์ชันนี้เพื่อให้ได้รูป polygon ดังตัวอย่างข้างล่าง กำหนดให้รัศมีของ polygon = 1 สำหรับงานนี้



**ตัวอย่างงานที่เสร็จแล้ว**



# Assignment-3: Change the Polygon’s Radius

สำหรับ Assignment-3 เป็นงานที่ทำต่อเนื่องมาจาก Assignment-1 และ Assignment-2 โดยให้นักศึกษาเขียนโค้ด JavaScript ที่ใช้ library Three.js เพื่อ**สร้าง Polygon ที่มีจุดศูนย์กลางของ polygon อยู่ที่พิกัด (x, y, 0) และมีรัศมี R โดยที่ x และ y เป็นค่าพิกัดที่ผู้ใช้กำหนดมา และ R เป็นค่ารัศมีที่ผู้ใช้อินพุทเข้า** คำอธิบายเพิ่มเติมของ assignment นี้ ให้ดูจากวีดีโอข้างล่าง

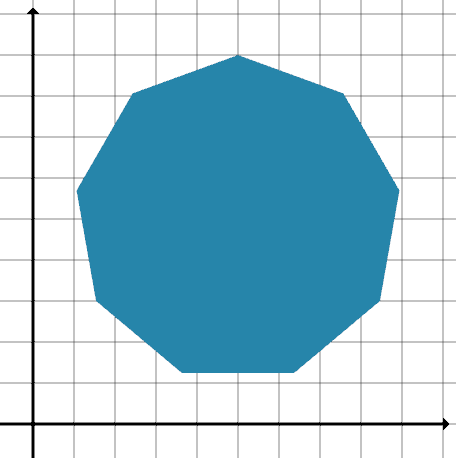
<https://www.youtube.com/v/D6ehAny6YpI>

**สิ่งที่นักศึกษาต้องทำ**

1. เขียน JavaScript ฟังก์ชัน ชื่อ **PolygonGeometry()** ซึ่งรับ parameter เป็นจำนวนด้านของ polygon (sides), พิกัดจุดศูนย์กลางของ polygon (location) และ รัศมี R (radius)
2. เมื่อรับค่าอินพุททั้งสามค่าสำหรับสร้าง polygon แล้ว ให้นักศึกษาสร้าง polygon โดยใช้จำนวน triangles ที่น้อยที่สุด
3. Template ของฟังก์ชัน **PolygonGeometry()** อยู่ในไฟล์ “***polygongeometry3.html***” นักศึกษาต้องเพิ่มโค้ดในฟังก์ชันนี้เพื่อให้ได้รูป polygon ดังตัวอย่างข้างล่าง



**ตัวอย่างงานที่เสร็จแล้ว**

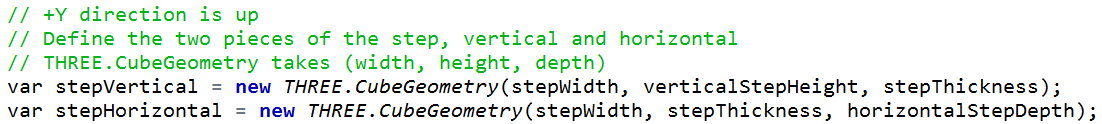


# Assignment-4: Build a Stairway

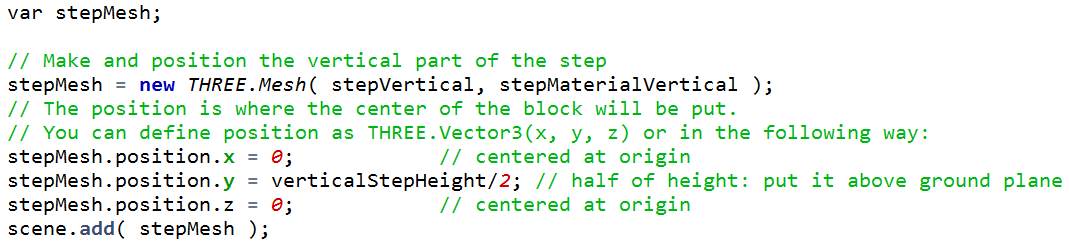
This assignment gets you familiar with how most of three.js’s geometric objects work. You’ll be spending your time working inside the create stairs function. See instruction’s video in the following link.

<https://www.youtube.com/v/8B14upf_WxQ>

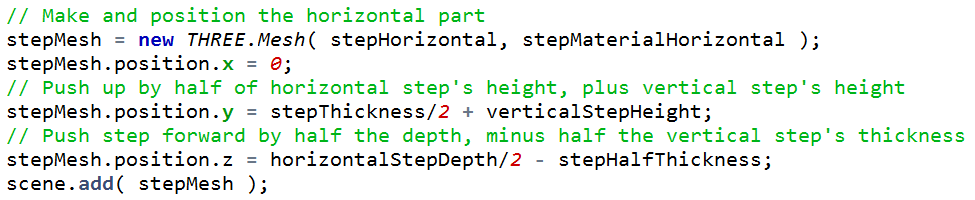




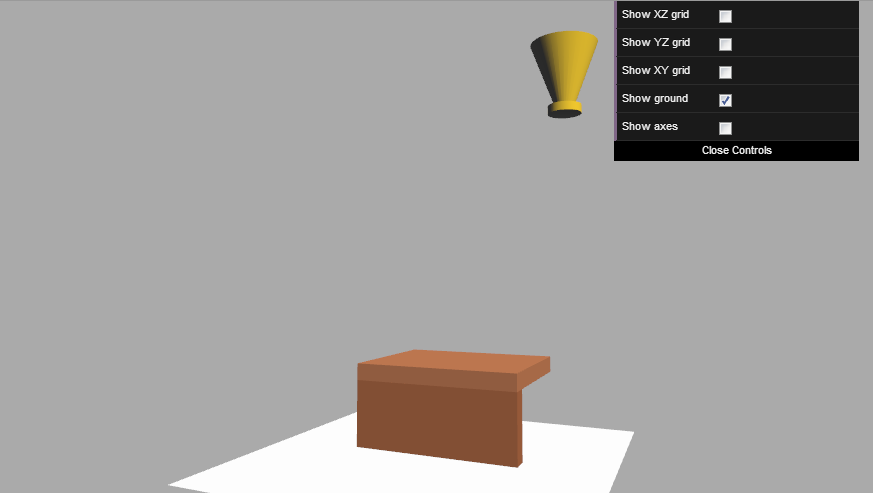
We’re going to use the CubeGeometry method to create two pieces for steps, a vertical piece and a horizontal plank. CubeGeometry is not a perfect name, since a cube is defined as having all its dimensions be the same size - BoxGeometry would have been better, but no big deal. Here’s the code for defining the geometry for our two steps. The way Cube Geometry works is that you provide it a size in X, Y, and Z. It then creates a set of triangles for you forming a box with these dimensions.



Once you have this geometric object, you can use it as many times as you wish to create three.js Meshes, each of which consist of geometry and a material. You can use the “position” parameter on the Mesh class to reposition the box, and in fact you’ll need to do this for the exercise. When you create a box, it is centered around the origin. For our vertical part of the step, we want this piece to be resting on the ground. To do so, we take the height of the piece and move it half this distance upward, so that the bottom is now at ground level.



The horizontal piece making up the flat part of the step is created with this code. In this case we first push the step up half its thickness, so it rests on the plane, then push it up the entire height of the vertical piece so that it rests on top. The z position is also changed, pushing the plank so that one edge is at the origin, then pulling it a bit back so it rests firmly on the vertical piece.



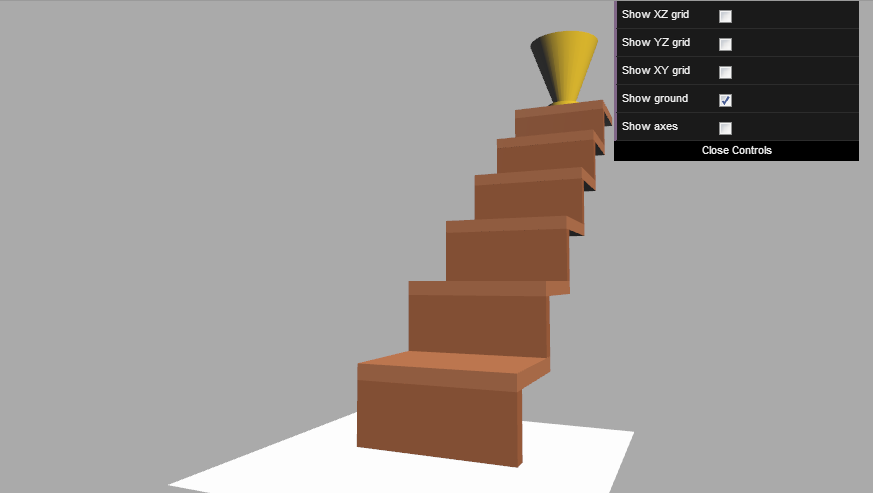
Once you run the program-template you’ll see the two step pieces, with the vertical piece resting on the ground plane. You’ll also notice a cool program feature: there are some controls available. The grids can help you get a sense of distance along each axis. They have lines every 100 units. Each is labeled with the axes that are in the grid plane. For example, since the Y axis is up for this scene, the XZ grid is on the ground plane. [ toggle this on ]

Your goal is to build a number of steps, one on top of the other going forward, that just reaches the gold cup.

**สิ่งที่นักศึกษาต้องทำ**

1. ทำการแก้ไขฟังก์ชัน ชื่อ **createStairs()** ให้ทำการสร้างบันไดต่อจากฐานไปจนถึงถ้วย
2. Template ของฟังก์ชัน **createStairs** อยู่ในไฟล์ “***buildstair.html***” โดยในฟังก์ชันดังกล่าว ได้ทำการสร้างบันไดมา 1 ขั้นเป็นตัวอย่าง นักศึกษาต้องเพิ่มโค้ดเพื่อให้ได้บันไดต่อขึ้นไป

**ตัวอย่างของงานที่เสร็จแล้ว**

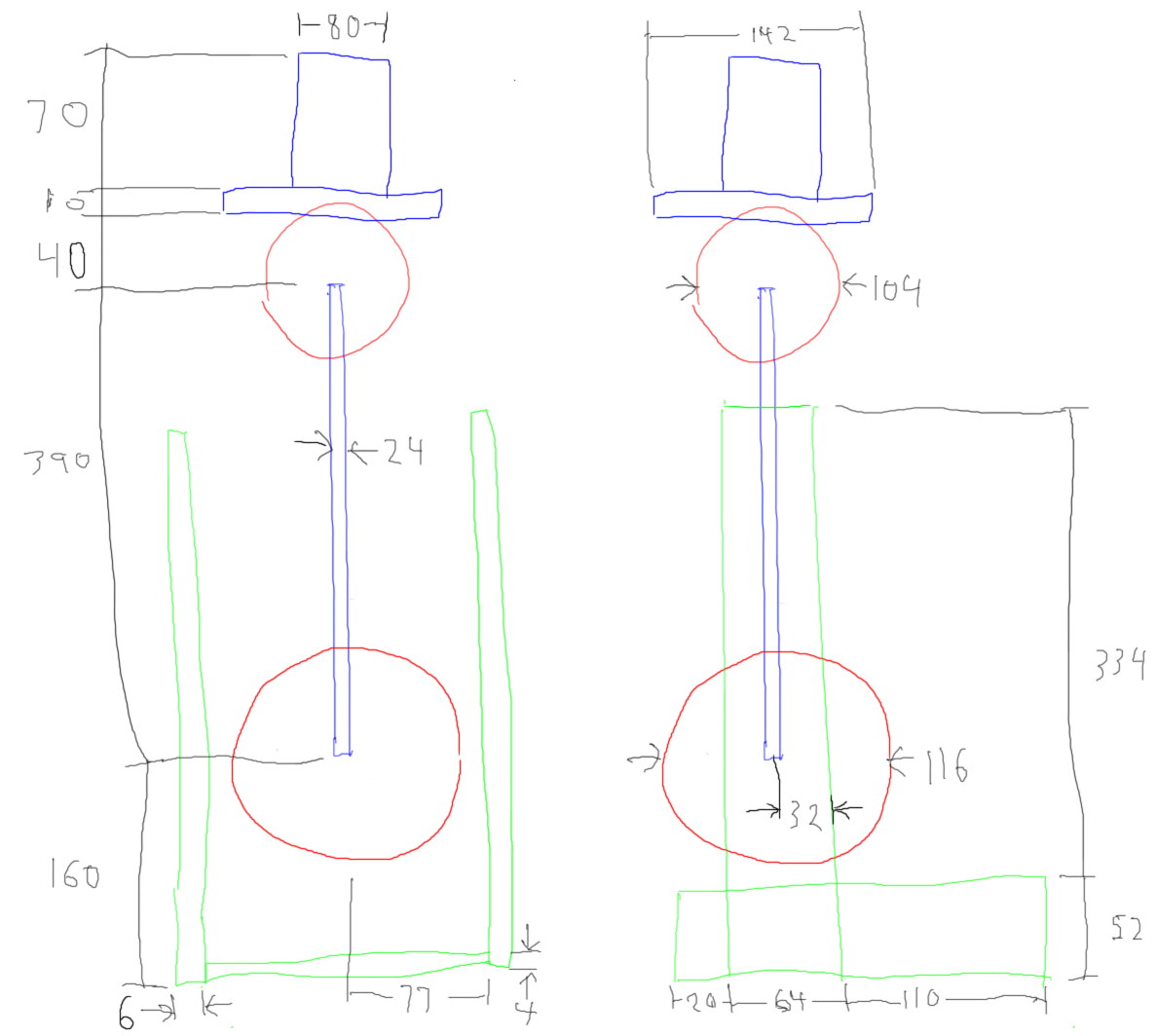


# Assignment-5: Make a Drinking Bird’s Body



Your goal is to make a simple model to manipulate in future units. You’ll create a basic drinking bird model - very basic, at this point he won’t even have a beak. We’ll get to that in a later unit. The drinking bird is a little toy that drinks from a cup - Google it!

<https://www.youtube.com/v/k9_PV3FtFbo>



The objects you’ll make are the legs, body, head (without a nose or eyes), and jaunty hat. Here’s a blueprint, showing front and side views. Well, it’s more a back-of-the-napkin drawing, but sometimes that’s all you get from a client. Me, my favorite client comment I heard that one designer received was, “could you please make that sphere rounder?”

radius, segmentsWidth, segmentsHeight

radiusTop, radiusBottom, height, radiusSegments ]

boxGeometry = new THREE.CubeGeometry( 125.6, 389.8, 202.1 );

sphereGeometry = new THREE.SphereGeometry( 202.1, 32, 16 );

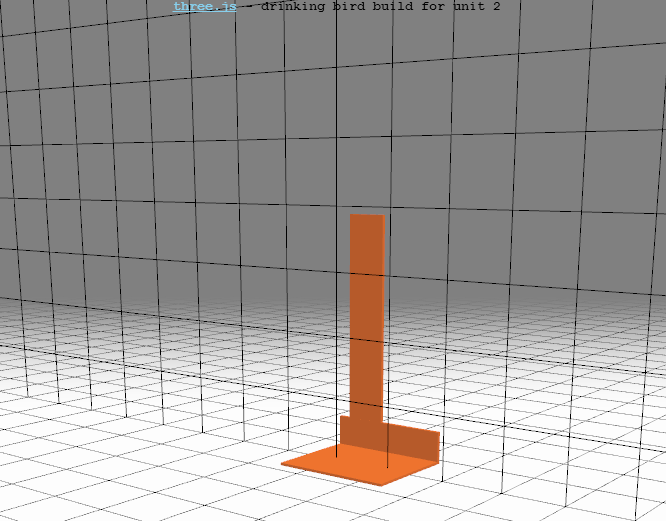
cylinderGeometry = new THREE.CylinderGeometry( 29.4, 202.1, 553.5, 32 );

The three objects you’ll use are the cube, sphere, and cylinder. Here are typical calls for creating these objects:

For the cube, the numbers passed in are the width, height, and depth of the box formed.

The sphere needs just a radius. The additional two numbers specify how much tessellation happens along the equator and from pole to pole, respectively. You might try changing these tessellation numbers to see the effect on the body, but please use these numbers 32 and 16 when you submit your answer.

The cylinder gives an upper and lower radius, followed by a height and by the amount of tessellation around its equator. Again, please use the number 32.



**Things you have to do**

* Modify code-template “***drinkingbird.html***”
* There are **three functions** you’ll be working on …
  + The function **createSupport()** will make the drinking bird’s support frame - base, legs and feet.To get you started, I’ve laid down a base that the legs connect to, along with the left leg. You have to add the right leg and feet.
  + The function **createBody()** will create the bird’s body and spine.
  + The function **createHead()** will make the head and hat.

You have to create the objects in the correct location and add them to scene. When you have one that, you will see the model of the drinking bird, just like this

