# THƯ VIỆN 3D THREE.JS

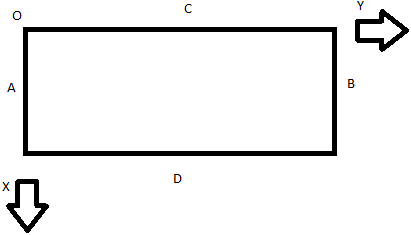
## Mô tả vấn đề

### Dữ liệu đầu vào

#### Dữ liệu cấu hình kho lạnh gồm

Là một đối tượng có ba thuộc tính:

* size: là đối tượng quy định kích thước của kho lạnh, gồm bốn thuộc tính:
  + x: chiều rộng của kho lạnh.
  + y: chiều dài của kho lạnh.
  + z: chiều cao của kho lạnh.
  + tilesize: kích thước của một đơn vị thể hiện trên mô hình, đơn vị là điểm ảnh.
* door: là đối tượng cấu hình cửa, gồm hai thuộc tính:
  + show: quyết định có hiển thị cửa trên mô hình hay không. Nếu show bằng true thì hiển thị cửa trên mô hình, ngược lại thì không hiển thị.
  + direction: Mô tả vị trí cửa theo hình bên dưới



* axis-labels: mô tả các nhãn được gắn trên ba trục tọa độ.
  + axis-x: mô tả các nhãn trên trục x.
    - show: xác định xem có hiển thị các nhãn trên trục x hay không.
    - list: mảng các số nguyên chứa tọa độ của các nhãn trên trục x.
  + axis-y: mô tả các nhãn trên trục y.
    - show: xác định xem có hiển thị các nhãn trên trục y hay không.
    - list: mảng các số nguyên chứa tọa độ của các nhãn trên trục y.
  + axis-z: mô tả các nhãn trên trục z.
    - show: xác định xem có hiển thị các nhãn trên trục z hay không.
    - list: mảng các số nguyên chứa tọa độ của các nhãn trên trục z.

#### Dữ liệu nhiệt độ:

Là một đối tượng có ba thuộc tính:

* values: là một mảng ba chiều chứa thông tin nhiệt độ của kho lạnh.
* min: nhiệt độ thấp nhất trong phòng.
* max: nhiệt độ cao nhất trong phòng

#### Dữ liệu cấu hình lát cắt:

Là một đối tượng có hai thuộc tính:

* axis: chứa tên trục muốn cắt, là một ký tự viết thường (x, y hoặc z).
* level: chứa một số nguyên, là số lớp muốn cắt.

### Kết quả đầu ra mong muốn

* Hiển thị khối hình hộp với các ô màu tương ứng với dữ liệu nhiệt độ đã cung cấp trước đó.
* Có thể xoay 360 độ, không có ô nào bị che khuất.
* Có thể cắt lát theo ba trục x, y và z mà không gặp vấn đề gì.
* Có thể hiển thị các nhãn đã cài đặt trên ba trục tọa độ, vị trí chính xác, nội dung rõ ràng.

## Giới thiệu thư viện three.js

Three.js là một thư viện JavaScript 3D dễ sử dụng, gọn nhẹ. Nó cung cấp nhiều trình kết xuất như canvas, WebGL, v.v.

WebGL (Thư viện đồ họa Web) là một API JavaScript mang đồ họa 3D được tăng tốc phần cứng cho trình duyệt mà không cần sự trợ giúp của các tiện ích bổ sung nào khác. WebGL chạy trên GPU (Bộ xử lý đồ họa) của card đồ họa của bạn, điều đó có nghĩa là CPU của bạn (Bộ xử lý trung tâm) có thể tập trung vào các tác vụ khác. Từ quan điểm của người dùng, các ứng dụng chỉ đơn giản là chạy nhanh hơn nhiều. Thư viện Three.js giúp làm việc với WebGL dễ dàng hơn.

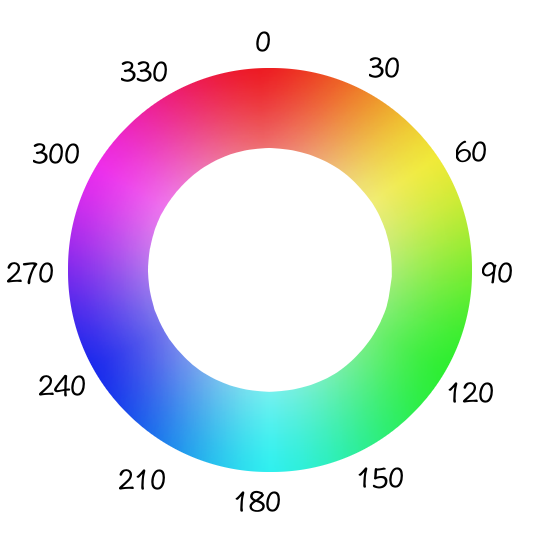
## Đối tượng sử dụng

…

## Hệ màu HSL

Không gian màu HSL, còn gọi là không gian màu HSV hay HSB, là một không gian màu dựa trên ba số liệu:

* H: (Hue) Vùng màu hay còn gọi là góc độ màu.
* S: (Saturation) Độ bão hòa màu.
* L (hay V, B): (Lightness, Value hay Bright) độ sáng.

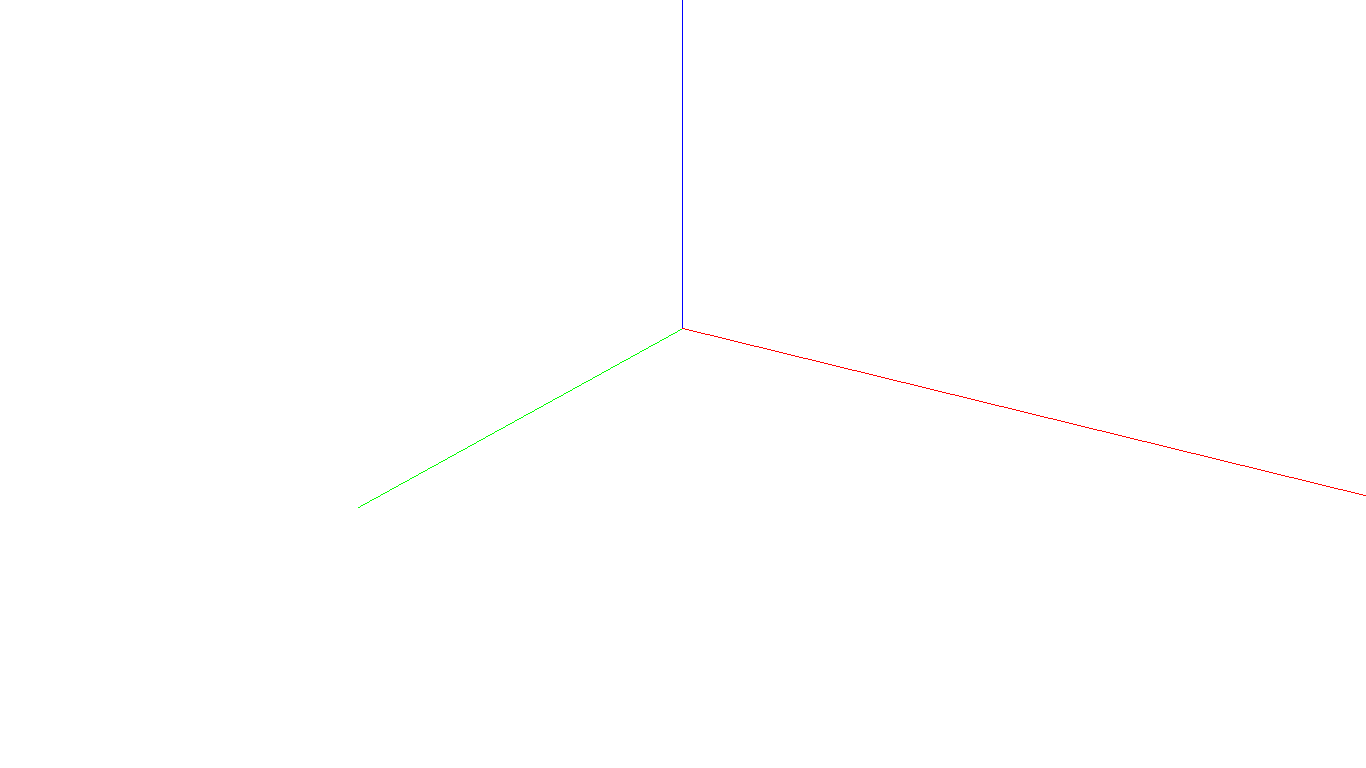


## Các bước dựng mô hình phân bố nhiệt độ trong không gian 3 chiều.

### Khởi tạo không gian 3 chiều

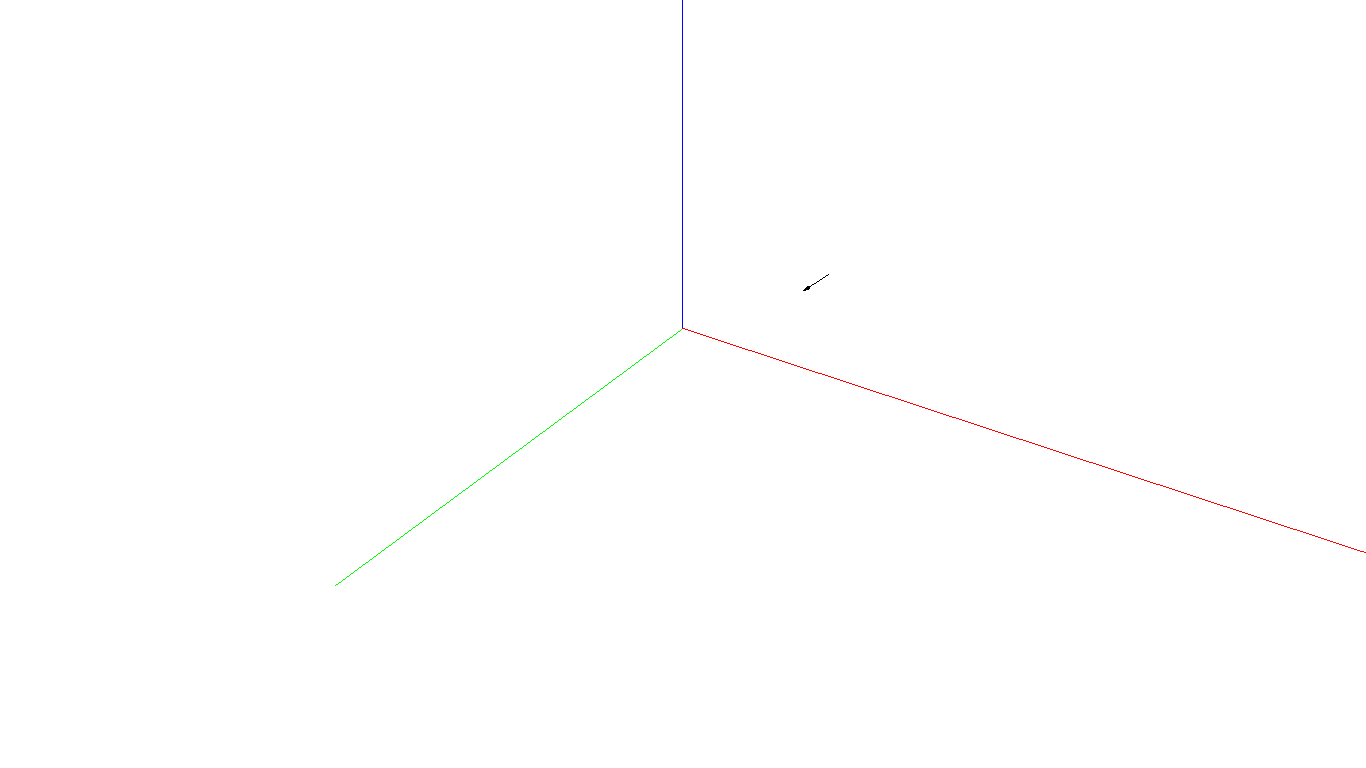
Khởi tạo các đối tượng cơ bản cần có của không gian ba chiều trong thư viện Three.js như:

* Scene: màn ảnh, chứa tất cả các đối tượng 3D bên trong.
* Camera: máy quay, chiếu góc hình lên màn ảnh, cho ra hình ảnh tại một góc nhìn xác định.
* Renderer: đối tượng vẽ hình chiếu ra bên ngoài.
* OrbitControls: cung cấp các tính năng điều khiển, xoay góc nhìn xung quanh mô hình.
* AxesHelper: vẽ các trục tọa độ lên màn ảnh.



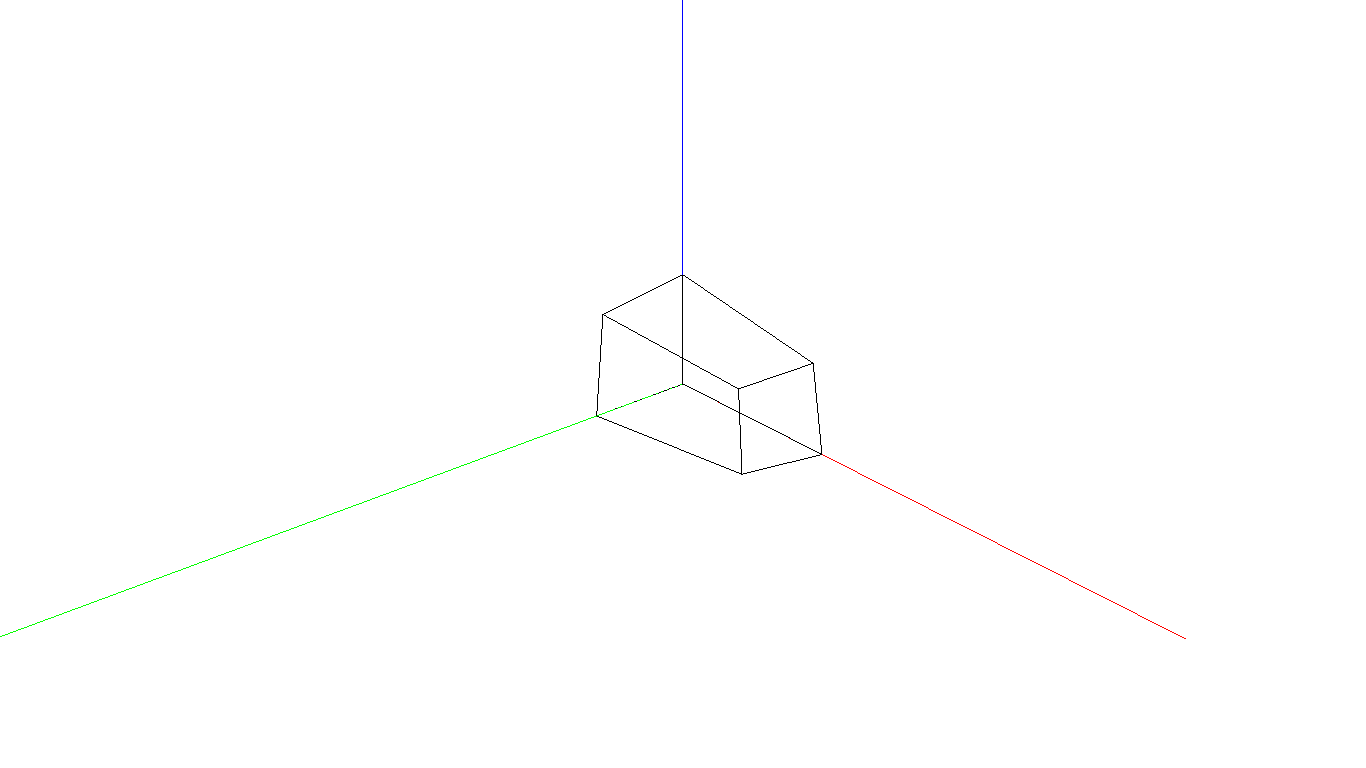
### Đánh dấu vị trí cửa

Đọc cấu hình được truyền vào, tính toán độ lớn và vị trí cửa. Tạo đối tượng ArrowHelper với các thông số đã tính toán để thể hiện vị trí cửa.



### Dựng khung giới hạn kích thước kho lạnh

Đọc cấu hình được truyền vào, tính toán kích thước, vị trí các nút và dựng các đoạn thẳng tại các vị trí thích hợp dựa trên dữ liệu đã đọc.



### Tính kích thước của khối dữ liệu cần hiển thị

Đọc cấu hình được truyền vào, bao gồm cấu hình lát cắt từ đó tính ra kích thước mới .

### Tạo đối tượng chứa khối dữ liệu

Thực hiện các công việc sau:

* Căn chỉnh trung tâm của máy quay tại trọng tâm của khối hộp.
* Tạo các khung mặt phẳng của khối hộp bằng phương thức createAFace.
* Gộp sáu đối tượng đã tạo vào một đối tượng Object3D.
* Thêm đối tượng Object3D vừa rồi vào màn ảnh.

#### Tạo 1 mặt phẳng của khối dữ liệu (createAFace)

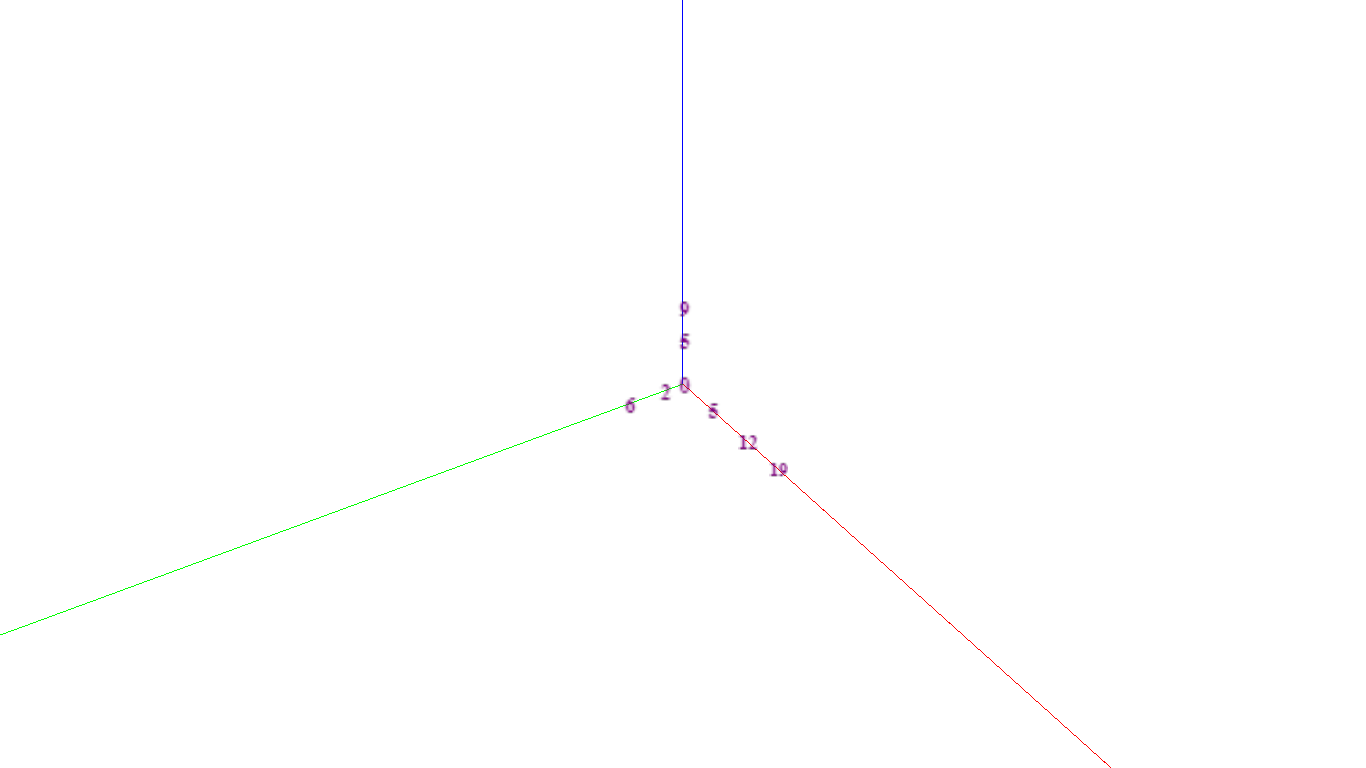
Gồm các bước:

* + Xác định kích thước khối hộp sắp tạo.
  + Xác định thứ tự của mặt cần tạo.
  + Xác định tọa độ, kích thước và góc quay dựa trên kích thước và thứ tự.
  + Tạo hình dạng của mặt phẳng.
  + Tạo vật liệu của mặt phẳng.
  + Tạo đối tượng vật thể của mặt phẳng.
  + Loại bỏ các đối tượng không cần thiết
  + Trả về vật thể vừa tạo.

### Dán các nhãn cần thiết lên các trục tọa độ

Gồm .. bước:

* Đọc dữ liệu cấu hình
* Duyệt qua các tọa độ cần gắn nhẵn
  + Tạo đối tượng Sprite (luôn đối mặt với camera).
  + Tính toán tọa độ.
  + Thêm đối tượng Sprite vừa tạo vào tọa độ thích hợp.



### Cập nhật dữ liệu nhiệt độ

Duyệt qua mảng dữ liệu, kiểm tra xem phần tử hiện tại có thuộc vào mặt phẳng nào của hình hộp không. Nếu có thì tính sắc độ và cập nhật màu của mô hình tại tọa độ đó. Nếu không thì bỏ qua.

#### Cập nhật dữ liệu nhiệt độ tại một tọa độ

Hàm nhận vào hai giá trị lần lượt là thứ tự mặt phẳng và nhiệt độ tại điểm cần cập nhật theo thứ tự từ trên xuống dưới, từ trái sang phải của mặt phẳng.

#### Chuyển nhiệt độ sang sắc độ

Hàm nhận vào ba giá trị lần lượt là:

* min: nhiệt độ thấp nhất.
* max: nhiệt độ cao nhất.
* temp: nhiệt độ của ô cần tính.

Giá trị trả về được tính theo công thức:

