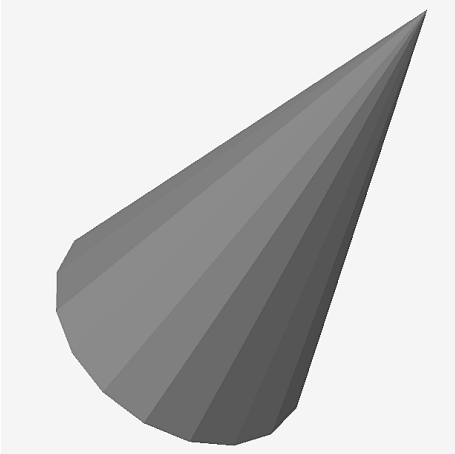
CAD Exchanger

Тестовое задание на позицию Javascript-разработчика

Задание нацелено на оценку уровня владения языком программирования, сопутствующих технологий и способности разобраться в новом материале при необходимости.
Задание составлено на английском языке, т.к. большая часть технической информации в будущем будет доступна именно на нем.
Пожалуйста, внимательно прочтите все обязательные требования! Вы также можете выполнить дополнительные задания (под «звездочкой»), это может стать Вашим преимуществом перед другими кандидатами.
Ответ следует направить на адрес jobs@cadexchanger.com, в виде ссылки на проект на github.
Дополнительно Вы можете разместить работающий веб-сайт с реализованным функционалом на Amazon, Heroku или любом другом ресурсе на Ваш выбор.

Create a simple client-server web application which can display a simple cone in a 3D view:



The application should provide the following functionality:

1. This should be a single page application (SPA), both with front-end and back-end.
2. The user should enter parameters (cone height, radius and number of segments on a circle) via client.
3. The client should transfer the data to the server.
4. The server must compute triangulation of the cone (i.e. a set of triangles to be used for display) and pass it back to the client.
5. The client should display computed triangulation in a 3D view using WebGL (e.g. with the help of the three.js library).
6. Make sure your implementation neither exposes memory leaks nor overutilizes CPU on repeated usage.

For back-end implementation we recommend using node.js (although you may choose any other alternative).

Tips: "How to compute cone triangulation"

To display a 3D object in a 3D view you will need its triangulated representation (i.e. the 3D object must be represented in the form of triangles).

A cone can be easily represented with N triangles (where N is a number of segments along its circle). Each triangle is composed of three nodes (A, P_i, P_{i+1}) – see fig.1.

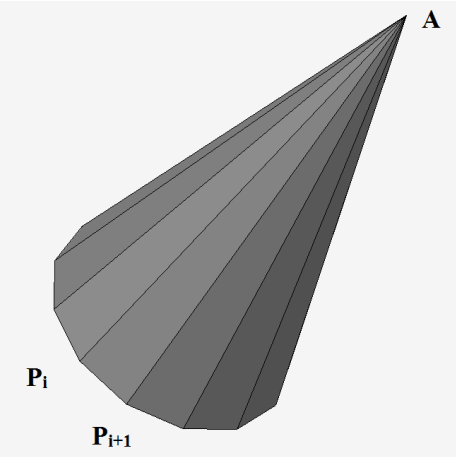


Fig.1. Triangulation of a cone

X, Y, Z coordinates of a node:

A: {0, 0, H}, where H is a cone height

P_i : {R * cos (2π * i / N), R * sin (2π * i / N), 0} , i=0,N-1 – see fig.2:

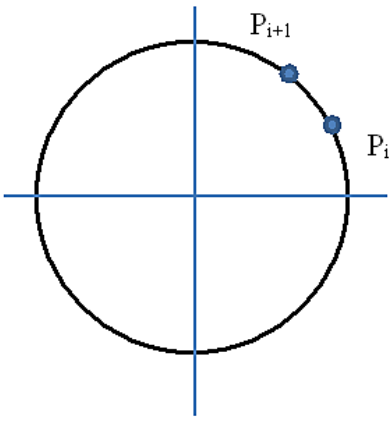


Fig.2 – triangulation nodes along the cone's circle

*** Additional assignments** (you can implement any or both of them)**:**

1. Implement smooth representation of a cone (see fig.3). For that you will need to compute a unit surface normal at each triangulation node (in addition to its coordinates) and pass it to the WebGL library. A unit surface normal on a conical surface is :

   $n_i = N_i / |N_i|$, where N_i is a vector (B, P_i), where B is a point on a cone axis – see fig.4.

   Coordinates of the point B:
   B: {0, 0, - R^2/H}.

Magnitude of the vector Ni: $|Ni| = \sqrt{(Pi.x - B.x)^2 + (Pi.y - B.y)^2 + (Pi.z - B.z)^2}$

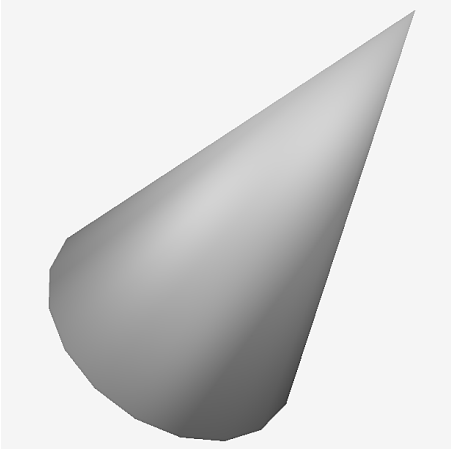


Fig.3. Smooth representation of a cone

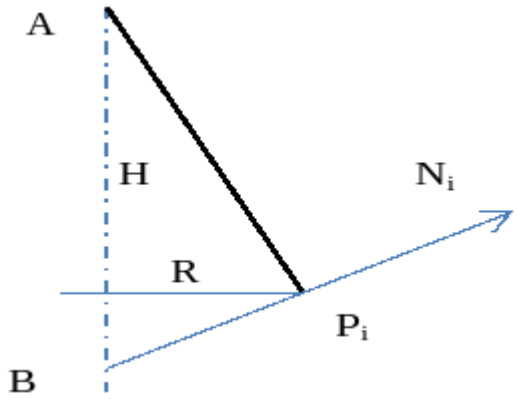


Fig.4. Tips to compute conical surface normal

2. Implement triangulation computation in a small self-written C#/Java/Python/C/C++ library (instead of a Javascript library) deployed on a server.