

Operación Fuego de Quásar

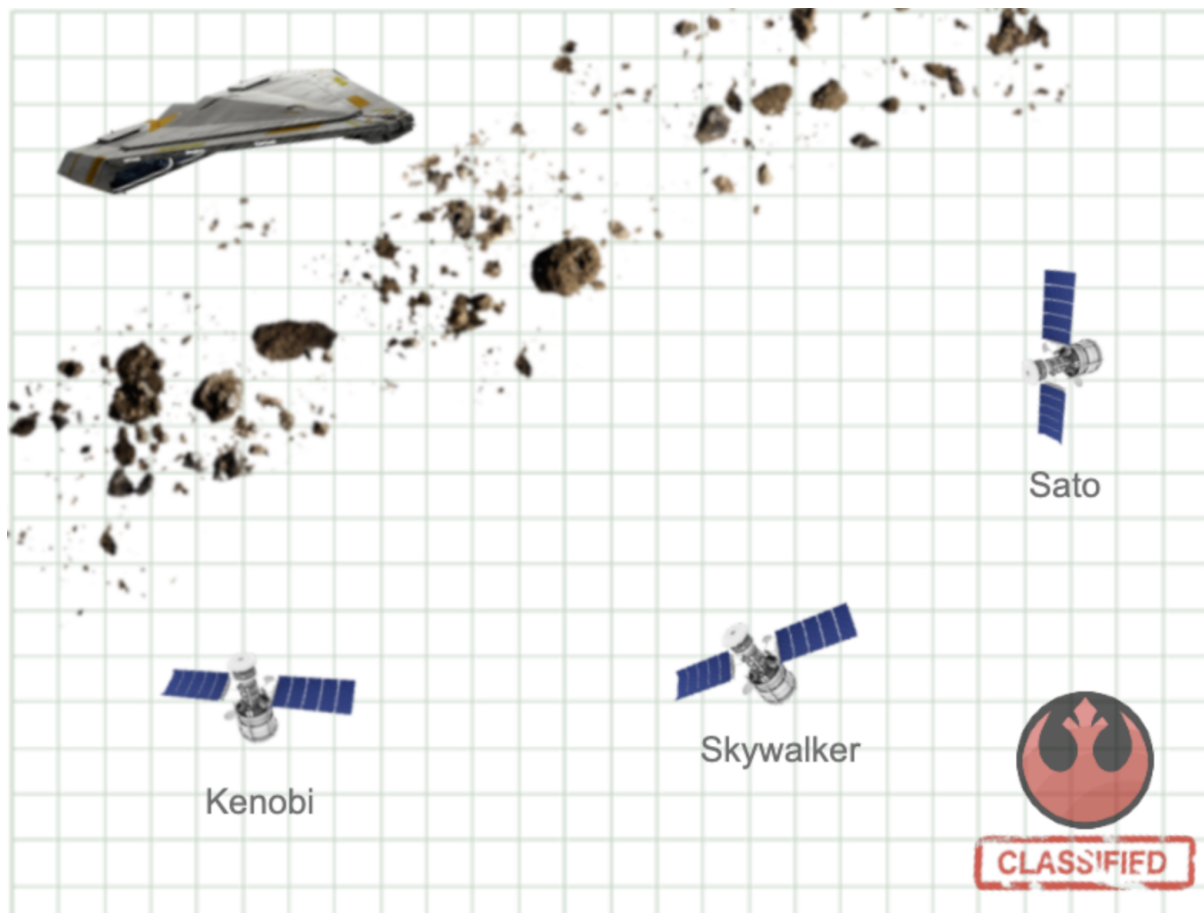


Evaluación para desarrolladores backend

Contexto

Han Solo ha sido recientemente nombrado General de la Alianza Rebelde y busca dar un gran golpe contra el Imperio Galáctico para reavivar la llama de la resistencia.

El servicio de inteligencia rebelde ha detectado un llamado de auxilio de una nave portacarga imperial a la deriva en un campo de asteroides. El manifiesto de la nave es ultra clasificado, pero se rumorea que transporta raciones y armamento para una legión entera.



Operación Fuego de Quásar



Evaluación para desarrolladores backend

Desafío

Como jefe de comunicaciones rebelde, tu misión es crear un programa en Java que retorne la fuente y contenido del mensaje de auxilio. Para esto, cuentas con tres satélites que te permitirán triangular la posición, ¡pero cuidado! El mensaje puede no llegar completo a cada satélite debido al campo de asteroides frente a la nave.

Posición de los satélites actualmente en servicio

- Kenobi: [-500, -200]
- Skywalker: [100, -100]
- Sato: [500, 100]

Nivel 1

Crear un programa con las siguientes firmas:

```
// input: distancia al emisor tal cual se recibe en cada satélite
// output: las coordenadas 'x' e 'y' del emisor del mensaje
Point getLocation(double[] distances)

// input: el mensaje tal cual es recibido en cada satélite
// output: el mensaje tal cual lo genera el emisor del mensaje
String getMessage(String[][] messages)
```

Consideraciones

- La unidad de distancia en los parámetros de *getLocation* es la misma que la que se utiliza para indicar la posición de cada satélite.
- El mensaje recibido en cada satélite se recibe en forma de arreglo de strings.
- Cuando una palabra del mensaje no pueda ser determinada, se reemplaza por un string en blanco en el array. Ejemplo: ["este", "es", "", "mensaje"]
- Considerar que existe un desfase (a determinar) en el mensaje que se recibe en cada satélite. Ejemplo:
 - Kenobi: ["", "este", "es", "un", "mensaje"]
 - Skywalker: ["este", "", "un", "mensaje"]
 - Sato: ["", "", "es", "", "mensaje"]

Operación Fuego de Quásar



Evaluación para desarrolladores backend

Nivel 2

Crear una API REST, con un servicio */topsecret* en donde se pueda obtener la ubicación de la nave y el mensaje que emite.

El servicio recibirá la información de la nave a través de un HTTP POST con un payload con el siguiente formato:

```
POST -> /topsecret/
{
  "satellites": [
    {
      "name": "kenobi",
      "distance": 100.0,
      "message": ["este", "", "", "mensaje", ""]
    },
    {
      "name": "skywalker",
      "distance": 115.5,
      "message": ["", "es", "", "", "secreto"]
    },
    {
      "name": "sato",
      "distance": 142.7
      "message": ["este", "", "un", "", ""]
    }
  ]
}
```

La respuesta deberá ser 200 con el siguiente body:

```
{
  "position": { "x": -100.0, "y": 75.5 },
  "message": "este es un mensaje secreto"
}
```

En caso que no se pueda determinar la posición o el mensaje, deberá ser un 404

Operación Fuego de Quásar



Evaluación para desarrolladores backend

Nivel 3 (Bonus)

Considerar que el mensaje ahora debe poder recibirse en diferentes POST al nuevo servicio */topsecret_split*, respetando la misma firma que antes. Por ejemplo:

```
POST /topsecret_split/{satellite_name}
{
  "distance": 100.0,
  "message": ["este", "", "", "mensaje", ""]
}
```

Crear un nuevo servicio */topsecret_split/* que acepte POST (para actualizar la posición de cada satélite)

Y un GET que deberá indicar la posición y el mensaje en caso que sea posible determinarlo y tener la misma estructura del ejemplo del Nivel 2. Caso contrario, deberá responder un mensaje de error indicando que no hay suficiente información.

Operación Fuego de Quásar



Evaluación para desarrolladores backend

Entregables

- Código fuente en repositorio privado de GitHub
- Documentación que indiquen cómo ejecutar el programa
- Documentación y diagramas del proyecto que considere importante
- Contemplar buenas prácticas (tip: imaginar que estas poniendo una aplicación productiva)