Documentación API

Introducción

Nuestra API consiste en un juego de aventuras multijugador donde los jugadores tendrán que administrar un ejército (que se moverá como una sola unidad por un mapa) y recursos para lograr conquistar el mayor número (con respecto a los otros jugadores) de poblados dentro del mapa antes de que se terminen los turnos de la partida. El ejército del jugador podrá ir mejorando al reclutar nuevos integrantes a este dentro de los poblados utilizando monedas o mediante accesorios. Además tendrá que procurar triunfar en enfrentamientos con otros jugadores y sobrepasar los peligros que pueden encontrarse dentro del mapa.

Comandos de la API WEB

Session/log_in:

- **Tipo**: POST
- **Función**: registrarse como usuario
- **Parámetros**:
    - {"username": nombre_usuario,
      "password": contraseña_usuario}

- **Retorno**:
    - Si existe ese usuario con esa contraseña se realiza el inicio de sesión mediante koa-session y se muestra que usuario se ha conectado. En caso de no existir, se muestra un mensaje de error

Session/log_out:

- **Tipo**: GET
- **Función**: Cerrar sesión
- **Retorno**:
    - En caso de existir un usuario conectado, se cierra la sesión y se muestra un mensaje en el *body* indicando que se ha cerrado la sesión del usuario correspondiente. Si no existe un usuario conectado, se muestra un mensaje de error.

Moves/process:

- **Tipo**: POST
- **Función**: Procesar el resultado de una jugada
- **Parámetros**:
    - Este archivo poseerá todos los datos actuales del jugador que realiza la jugada (´jugador') además de las acciones que realizó en la jugada

('jugada'), un arreglo que contiene identificadores para todos los lugares del mapa, el id de la partida y los turnos restante de la partida.

```
'jugador' : {
'username' : string , // Nombre de usuario
'id_jug' : int , // Número de jugador en la partida
'guerreros' : int , // Cantidad de guerreros
'doctores' : int , // Cantidad de doctores
'obreros' : int , // Cantidad de obreros
'dinero' : int , // Cantidad de monedas
'madera' : int , // kilos de madera
'accesorios' : Array , //Arreglo que contiene los
accesorios del jugador
'poblados' : int , //Número de poblados conquistados
'mapa_visitados' : Array , //Arreglo que contiene
casillas visitadas
'vida_actual' : int , // Vida actual
'vida_max' : int , // Vida máxima
'salud' : int , // Valor Stat de Salud
'ataque' : int , // Valor Stat de Ataque
'defensa' : int // Valor Stat de Defensa
},
'jugada' : {
'numero' : int , //Numero de turno
'x' : int , //columna resultante
'y' : int , //fila resultante
'interactuar' : int , //1: sí; 0: no
'curar' : int , //1: sí; 0: no
'conquistar' : int , //1: sí; 0: no
},
'mapa_lugares' : Array , //Arreglo que contiene los lugares
'id_partida' : int , // ID de la partida
'turnos_partida' : int //Número de turnos restantes
```

- **Retorno**:
    - Este archivo es muy similar al anterior pero contendrá los datos actualizados tras haberse procesado la jugada, además de otros nuevos.

```
'jugador' : {
'username' : string , // Nombre de usuario
'id_jug' : int , // Número de jugador en la partida
'guerreros' : int , // Cantidad de guerreros
'doctores' : int , // Cantidad de doctores
'obreros' : int , // Cantidad de obreros
'dinero' : int , // Cantidad de monedas
'madera' : int , // kilos de madera
'accesorios' : Array , //Arreglo que contiene los
accesorios del jugador
'poblados' : int , //Número de poblados conquistados
'mapa_visitados' : Array , //Arreglo que contiene
casillas visitadas
'vida_actual' : int , // Vida actual
```

'vida_max' : int , // Vida máxima
'salud' : int , // Valor Stat de Salud
'ataque' : int , // Valor Stat de Ataque
'defensa' : int // Valor Stat de Defensa
},
'jugada' : {
'numero' : int , //Nuevo número de turno
'x' : int , //Columna tablero tras procesar
'y' : int , //Fila tablero tras procesar
'exito_interactuar' : int , //1: exito, 0: fracaso y
null: no se realiza
'exito_conquistar' : int , //1: exito, 0: fracaso y
null: no se realiza
},
'mapa_lugares' : Array , //Arreglo que contiene los lugares
'id_partida' : int , //Id de partida
'turnos_partida' : int //Número de turnos restantes
'muere' : int //1: jugador muere; 0: jugador no muere};

users/id:

- **Tipo**: GET
- **Función**: Obtener la información de un usuario
- **Parámetros**:
  - {“user_id” : INT}
- **Retorno**:

({

username: STRING,

password: STRING,

token: STRING,

part_g: INTEGER,

part_p: INTEGER

}

users/new:

- **Tipo**: POST
- **Función**: Crear un usuario
- **Parámetros**:
  - JSON asociado a datos de un usuario a crear
    { username: STRING
    Contraeña: STRING }
- **Retorno**:
  - JSON asociado al usuario creado

({

username: STRING,

password: STRING,

token: STRING,

part_g: INTEGER,

part_p: INTEGER

}

users/id:

- **Tipo**: PATCH
- **Función**: Modificar la información de un usuario
- **Parámetros**:
  - Id de usuario a modificar
  - JSON con los datos nuevos
    { username: STRING
    Contraeña: STRING }

- **Retorno**:
  - JSON con los nuevos datos de aquel usuario

    ({

username: STRING,

password: STRING,

token: STRING,

part_g: INTEGER,

part_p: INTEGER

}

players/id:

- **Tipo**: GET
- **Función**: Obtener la información asociada a un jugador
- **Parámetros**:
  - Id único de algún jugador
- **Retorno**: JSON con la información del jugador

  ({

user_id: DataTypes.INTEGER,

username: DataTypes.STRING,

id_jug: DataTypes.INTEGER,

part_id: DataTypes.INTEGER,

guerreros: DataTypes.INTEGER,

doctores: DataTypes.INTEGER,

obreros: DataTypes.INTEGER,

```
dinero: DataTypes.INTEGER,
madera: DataTypes.INTEGER,
accesorios: DataTypes.ARRAY(DataTypes.STRING),
poblados: DataTypes.INTEGER,
mapa_visitados: DataTypes.ARRAY(DataTypes.ARRAY(DataTypes.STRING)),
vida_actual: DataTypes.INTEGER,
vida_max: DataTypes.INTEGER,
salud: DataTypes.INTEGER,
ataque: DataTypes.INTEGER,
defensa: DataTypes.INTEGER,
pos_x: DataTypes.INTEGER,
pos_y: DataTypes.INTEGER
}
```

-

accesories/id:

- **Tipo**: GET
- **Función**: Obtener la información asociada a un accesorio
- **Parámetros**:
  - Id de algún accesorio
- **Retorno**:
  - JSON con todos los datos del accesorio

```
({
    nombre: DataTypes.STRING,
    bono_ataque: DataTypes.FLOAT,
    bono_salud: DataTypes.FLOAT,
    bono_defensa: DataTypes.FLOAT,
    bono_vida: DataTypes.INTEGER
}
```

Part_publicas/id:

- **Tipo**: GET
- **Función**: Obtener información asociada a una partida publica
- **Parámetros**:
  - Id de partida publica
- **Retorno**:
  - JSON con todos los datos de la partida

```
({
```

```
n_part: DataTypes.INTEGER,
id_part: DataTypes.STRING,
num_partidas: DataTypes.INTEGER,
n_jug: DataTypes.INTEGER,
mapa: DataTypes.ARRAY(DataTypes.ARRAY(DataTypes.STRING)),
jug: DataTypes.ARRAY(DataTypes.STRING)
}
```

Part_publicas/new:

- **Tipo**: POST
- **Función**: crear una nueva partida publica
- **Parámetros**:
  - JSON con los datos de la partida a crear

```
({
    n_part: DataTypes.INTEGER,
    id_part: DataTypes.STRING,
    num_partidas: DataTypes.INTEGER,
    n_jug: DataTypes.INTEGER,
    mapa: DataTypes.ARRAY(DataTypes.ARRAY(DataTypes.STRING)),
    jug: DataTypes.ARRAY(DataTypes.STRING)
}
```

- **Retorno**:
  - JSON asociado a la partida creada

```
({
    n_part: DataTypes.INTEGER,
    id_part: DataTypes.STRING,
    num_partidas: DataTypes.INTEGER,
    n_jug: DataTypes.INTEGER,
    mapa: DataTypes.ARRAY(DataTypes.ARRAY(DataTypes.STRING)),
    jug: DataTypes.ARRAY(DataTypes.STRING)
}
```

Players/new:

- **Tipo**: POST
- **Función**: Crear un usuario
- **Parámetros**:
  - {username, id_partida}
- **Retorno**:
  - Retorna JSON del usuario creado

```
({
```

username: DataTypes.STRING,
password: DataTypes.STRING,
token: DataTypes.STRING,
part_g: DataTypes.INTEGER,
part_p: DataTypes.INTEGER
}

moves/start:

- **Tipo**: POST
- **Función**: Crear un usuario
- **Parámetros**:
  - Recibe el username de un usuario en formato JSON {username: STRING}
- **Retorno**:
  - Retorna un JSON con todos los datos del jugador y de la partida asociada

```
'jugador' : {
'username' : string , // Nombre de usuario
'id_jug' : int , // Número de jugador en la partida
'guerreros' : int , // Cantidad de guerreros
'doctores' : int , // Cantidad de doctores
'obreros' : int , // Cantidad de obreros
'dinero' : int , // Cantidad de monedas
'madera' : int , // kilos de madera
'accesorios' : Array , //Arreglo que contiene los
accesorios del jugador
'poblados' : int , //Número de poblados conquistados
'mapa_visitados' : Array , //Arreglo que contiene
casillas visitadas
'vida_actual' : int , // Vida actual
'vida_max' : int , // Vida máxima
'salud' : int , // Valor Stat de Salud
'ataque' : int , // Valor Stat de Ataque
'defensa' : int // Valor Stat de Defensa
},
'jugada' : {
'numero' : int , //Nuevo número de turno
'x' : int , //Columna tablero tras procesar
'y' : int , //Fila tablero tras procesar
'exito_interactuar' : int , //1: exito, 0: fracaso y
null: no se realiza
'exito_conquistar' : int , //1: exito, 0: fracaso y
null: no se realiza
},
'mapa_lugares' : Array , //Arreglo que contiene los lugares
'id_partida' : int , //Id de partida
'turnos_partida' : int //Número de turnos restantes
```

'muere' : int //1: jugador muere; 0: jugador no muere};

moves/check_player:

- **Tipo**: POST
- **Función**: revisar si es el turno de un usuario
- **Parámetros**:
  - Recibe un username de un usuario
    {username: STRING}
- **Retorno**:
  - JSON {correct: 1 } si es el turno de aquel usuario y {correct:0} siesque no lo es.

moves/winner:

- **Tipo**: POST
- **Función**: Obtener informacion de jugador que va ganando
- **Parámetros**:
  - JSON con {id_partida: INT}
- **Retorno**:
  - Todos los datos del jugador que valla ganando o halla ganado.

```
({
    user_id: DataTypes.INTEGER,
    username: DataTypes.STRING,
    id_jug: DataTypes.INTEGER,
    part_id: DataTypes.INTEGER,
    guerreros: DataTypes.INTEGER,
    doctores: DataTypes.INTEGER,
    obreros: DataTypes.INTEGER,
    dinero: DataTypes.INTEGER,
    madera: DataTypes.INTEGER,
    accesorios: DataTypes.ARRAY(DataTypes.STRING),
    poblados: DataTypes.INTEGER,
    mapa_visitados: DataTypes.ARRAY(DataTypes.ARRAY(DataTypes.STRING)),
    vida_actual: DataTypes.INTEGER,
    vida_max: DataTypes.INTEGER,
    salud: DataTypes.INTEGER,
    ataque: DataTypes.INTEGER,
    defensa: DataTypes.INTEGER,
    pos_x: DataTypes.INTEGER,
    pos_y: DataTypes.INTEGER
```

moves/end:

- **Tipo**: POST
- **Función**: Eliminar un jugador de una partida y si es el ultimo jugador, se elimina la partida.
- **Parámetros**:
  - JSON
    {username: STRING}
- **Retorno:**
  - Retorna los datos del jugador que ha abandonado la partida.

```
({
    user_id: DataTypes.INTEGER,
    username: DataTypes.STRING,
    id_jug: DataTypes.INTEGER,
    part_id: DataTypes.INTEGER,
    guerreros: DataTypes.INTEGER,
    doctores: DataTypes.INTEGER,
    obreros: DataTypes.INTEGER,
    dinero: DataTypes.INTEGER,
    madera: DataTypes.INTEGER,
    accesorios: DataTypes.ARRAY(DataTypes.STRING),
    poblados: DataTypes.INTEGER,
    mapa_visitados: DataTypes.ARRAY(DataTypes.ARRAY(DataTypes.STRING)),
    vida_actual: DataTypes.INTEGER,
    vida_max: DataTypes.INTEGER,
    salud: DataTypes.INTEGER,
    ataque: DataTypes.INTEGER,
    defensa: DataTypes.INTEGER,
    pos_x: DataTypes.INTEGER,
    pos_y: DataTypes.INTEGER
    }
```