## Role-Playing Game API

You are given a role-playing game API that gives the base for creating a game with characters, resources, battles and resource gathering. The API includes an engine, which controls the game world and executes commands on the objects within it. You are also given an Angular project which processes commands from the input.

There are some simple rules the API supports:

* Objects can be created anywhere
* Objects can go anywhere if they support the “go” command
* Objects can attack if they are fighters
  + Attacks always have single targets – one object can attack at most one other object at a time
* Objects can gather resources if they are gatherers
* Objects can be gathered if they are resources
* Objects have owners. An owner can be interpreted as “the team to which the object belongs”
* An object can be neutral

### Important Classes and Interfaces

These are the API’s interfaces:

* WorldObject – an abstract class that that has the following properties: **isDestroyed**, **hitPoints**, **position**, **canMove**, **team**. Team can be Blue, Red or Neutral. The **canMove** property should be **true** by default (if not provided).
* Resource – extends WorldObject class and additionaly has quantity and type. Quantity is always equal to hitPoints. The team is always neutral, and the object cannot move. There are 3 types of resources – Food, Lumber and Iron
* Unit – extends WorldObject class and adds the following properties: name, attack, defense, canGather and type. A unit can move. A unit can be either a Lumberjack or a Guard. The Lumberjack has 10 attack, 10 defense, can gather resources and has 20 hit points. The Guard has 20 attack, 5 defense, cannot gather resources and has 100 hit points.

### Commands

There are two types of commands the Engine supports:

* Object creation command – creates an object in the world
  + Format: create <object-type > <name> <coordinates> <type>
  + Example: create unit Kris 0,0 Blue Guard – creates a guard unit called Kris from the Blue team on position (0,0)
* Object action command – orders an object to execute a command. If the object can execute such a command, a string is printed, describing the result of the command. If not, a string is printed, notifying the inability of the object to execute the command.
  + Format: order <name> <command-name> <command-parameters>
  + Commands:
    - attack (no parameters)
    - gather (no parameters)
    - go <coordinates>
  + Example: order Kris attack – causes the Engine to search for an object with the Name Kris and makes it attack other objects at the object’s coordinates, printing the corresponding strings for success or failure.
  + Example: order Kris gather - causes the Engine to search for an object with the Name Kris and attempts to gather the resource at the object’s coordinates, printing the corresponding strings for success or failure.
  + Example: order Kris go 42,53 – causes the Engine to search for an object with the Name Kris and moves it to coordinates 42, 53.
* Miscellaneous commands:
  + show <parameters>
  + Commands:
    - all – prints summarized information of all uinits and objects on the field.
    - units <team> - prints unit names and locations for the selected team
    - resources – prints location and quantity of all available resources
    - <coordinates> - prints who is on the selected coordinates (Example: “show 0,0”)

Here is a list of all commands the Engine needs to support:

* **create resource lumber <position> <size>** - creates lumber with the specified size at the specified coordinates
* **create unit <name> <coordinates> <owner> lumberjack** - creates a lumberjack with the specified name at the specified coordinates and belonging to the specified owner (player)
* **create unit <name> <coordinates> <owner> guard** - same as the previous, but creates a guard
* **order <name> go <coordinates>** -makes the object with the specified name go to the specified coordinates
* **order <name> attack** -orders the object with the specified name to attack another object at its coordinates
* **order <name> gather** -orders the object with the specified name to gather a resource at its coordinates

TBD: More commands, how the attack/gather works, summary for resources and units

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| **create unit Kris 0,0 Blue Guard** | **Guard Kris is now at position 0,0** |