Scripting Islands of Battle, Workshop 3

Refresh

In the last workshop we covered building and accessing the world through `for` loops as well as getting player data.

You should now have a script that tries to login and registers an account if it fails to do so. It should then place your Castle on a random tile in the world and try to build a House on every tile you own (which will, at present, be only your Castle and so will never successfully build).

In this short third and final workshop we'll cover how the movement of units (and by extension fighting) works and suggest a few ideas on how a script might work or be structured.

Unit movement

To move units off of a tile use the command `move` with parameters `position` (the position that you're moving units *from*), `newPosition` (the position that you're moving units *to*), `number` (the number of units you're trying to move. If this is more than the amount on the tile it'll move all minus 1. You must always have at least 1 unit on a tile that you own) and your `authcode`.

For example, to move 15 units from tile 100 to tile 200 the command is `api.get("?a=move&position=100&newPosition=200&number=15&authcode="..authcode)`.

Battles

Battles are triggered when you attempt to move units onto a tile that is owned by an enemy. The current algorithm gives a 33% chance of an attacker causing between 1 and the number of units attacking kills, else a 1 to the a third of the number of units attacking kills. The attacker has a 50% chance of losing 1 to the number of units defending, else a 1 to the third of the number of units defending.

> You don't need to know this, really, but it might influence your tactical decisions. Defenders have an algorithmic advantage.

When attacking you'll get `attackerLosses`, `defenderLosses`, `attackingUnits` and

`defendingUnits`. So, for example, to print when an attack is dealt the code would be:

```
result = api.get(?
a=move&position=100&newPosition=200&number=15&authcode="..authcode)
if result.attackerLosses then -- a fight has occurred
  print(result.attackerLosses.."/"..fight.attackingUnits.." lost
"..fight.defenderLosses.."/"..fight.defendingUnits.." defeated.")
end
```

Task 6: modify your script so that once your Castle is placed your units move out in different directions.

You now know all you need to know to start building out your script! For further information refer to the API's documentation, available here: GitHub - Pebsie/APIBattle: A web service RTS that is comprised entirely of an API.

Ideas for structuring your scripts

It's essential that you code in registration and login scripts, as well as castle placement. Beyond that, it's up to you how your strategy works. Your script can be as complicated as you'd like it to be but remember that you're limited to 100 API calls a second.

- When do you move units out? How far out do they go? Which direction do you want to expand in? Does this influence the positioning of your Castle?
- Do you want to route out and play aggressively or will your script avoid enemy units until you've reached a certain number of units? How about collecting all of your units on a single tile then moving that single one out?
- You have the full capabilities of Lua scripting at your disposal: functions, plugins… the potential for expansion is limitless. The more you understand of Lua the more advanced things can be.