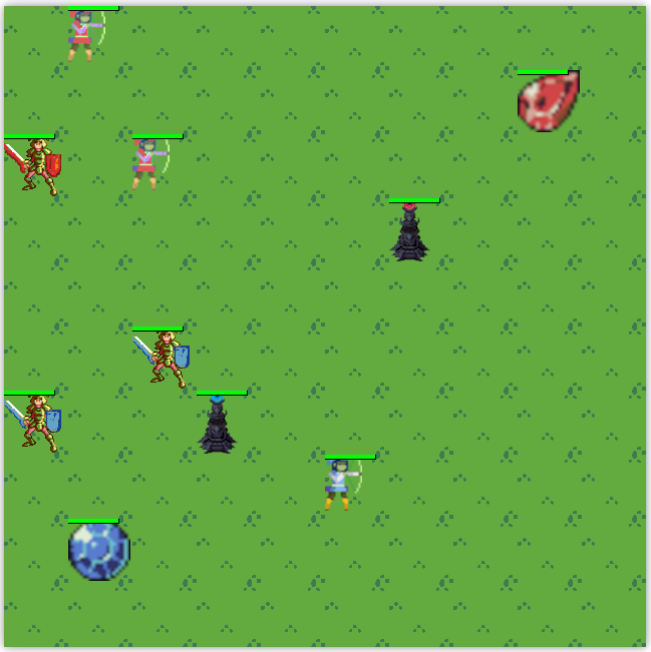
UNIVERSITÉ DU LUXEMBOURG

Lab 6: A Simple Artificial Intelligence for a MOBA game

Programming Fundamentals 2

20th May 2021



Goals

✯ Programming a simple artificial intelligence (AI) able to play a MOBA-like game.

- For this laboratory, all the AIs of all the students are public, so you can compare yourself to others.
- You can use the code of others, but you must credit the other student for it, and your contributions must be clear.
- Tournament during the last class on 27th of May.
- **Deadline: Monday 31th** May 08:00AM

Deliverables

1. The code on your Github repository generated by clicking here: `https://classroom.github.com/a/7D9NWNoV`

1 LOL 2D

After cloning the repository, you will have a copy of the LOL 2D game. You can create your own AI by extending an existing AI such as `lol/client/ai/RandomAI` or directly inheriting from `AIBase`. There are two main methods you need to override in your AI class:

```
public abstract Turn championSelect();
public abstract Turn turn();
```

The method `championSelect` is called only once at the beginning of the game in order to select your champions. The method `turn` will be called at each turn in order to record your *actions*. An action is, for instance, move a champion or attack a target (tower / nexus / champion). You must record your actions in an object of type `Turn` with the method `registerAction`. For instance, `turn.registerAction(new Attack(teamID, id, nexus.x(), nexus.y()));` registers an attack action for the champion with ID `id` to attack an enemy nexus. The server will execute the list of actions sequentially. A champion can only perform a single action per turn. If you register more than one action for a champion, the extra actions will be ignored. If an action is impossible, for instance attacking a target out of reach, the action is ignored by the server, and does not count as an action used by the champion.

You can check the existing AI in `lol/client/ai/` to create yours. An advice is to implement one idea at a time, and try to keep things simple and working.

If you modify something outside of `client`, you should submit a pull request (see lab 5). Indeed, if your version of the game is diverging from others, then you will not be able to compete against others. Everybody should have the same version of the game.

The game will probably be updated during the course of this laboratory, so you might want to update it on your side as well:

```
git pull https://github.com/ptal/lol2D.git
# Then fix (possible) conflicts in the code, then commit.
```