TP: TAMAGOTCHI MECHANICS

PABLO TESONE

1. Introduction

This TP aims to implement a non-trivial problem in Object-Oriented Programming. You will have to implement in Pharo or Java all the requirements included in this document.

2. Problem Context

Bandai, the creator of the Tamagotchi[1] game hires us to create a new version of the game.

We have to implement the behaviour logic behind the simulated pets. In this new version, we are going to have two Tamagotchis interacting with each other and performing some basic activities together.

Our virtual pet can be sad, hungry or happy. And its behaviour depends on its state. The state of the virtual pet changes only when an action is performed.

A new pet starts with a happiness level of 0 and it is in a happy state. If a pet is already in a given state, nothing should happen if it is required to change to that state (*e.g.* if it is happy, and I tell the pet to be happy, nothing should change in its state).

2.1. **Actions.**

Eating.

- If it is hungry, it gets happy.
- If it is happy, it increases its happiness level by one.
- If it is sad, it will not eat, and an error should happen.

Playing Alone.

- If it is hungry, its happiness level is reduced by 4.
- If it is happy, it increases its happy level by two. If it plays two times since eating, it gets hungry.
- If it is sad, it gets happy.

[1] `https://en.wikipedia.org/wiki/Tamagotchi`

Playing With Other Pet.

- If any of them is hungry, they don't play and nothing changes.
- If the pet is happy, it increases its happy level by four. If it plays two times since eating, it gets hungry.
- If any of them is sad, it gets happy.

3. Extensions

As the game gots so popular, we need to add new mechanics to it. We are going to add two additional types of Pets.

Dog.

- Whenever it eats, its happiness level rises to 5, then it rises normally by 1.

Lonely Cat.

- Whenever it plays with another pet, it gets sad.

4. Test Cases

It is required to implement a set of test cases covering all features. It is important to select what test cases to write, having excessive tests is not good at all.

Test cases should cover the different states and the possible actions.

For the added extensions is only needed to test the variation of behaviour, *e.g.* for the Lonely Cat is only needed to make it play.

5. Administrative Considerations

The TP should be sent by the 27/10/2023. Any later sending will be not taken into account. The TP should be sent as a single zip file with the content of the project.

For Pharo solutions, you can export the whole package doing right-clicking and selecting 'File Out' generating a single file with all the code of the package, that is the file to put in the Zip.

For Java solutions, it is important to Zip the entire project. You might include any other document you consider important to explain your solution, *e.g.* if you have taken any special consideration or understanding of a given point.

In all the cases, the delivered code should compile and work. Non-compiling code will be automatically rejected.

6. About Doubts and Different Interpretations

All the rules, descriptions and requirements on this document are open to different understanding. If a point is unclear or it raises different alternatives, you are free to choose the one that better suits you. The only condition is to explain in the affected part of the TP that you have taken a given direction and why you have done it.