COEN 359 Design Patterns

Project: Tetra-Star Simulation **Teams of 2 allowed**

Tetra-Star Simulation is a game-like **simulation project** (that can be enhanced and developed into a full-fledged game in future versions).

You are required to design and implement it in Java using OO Design Patterns where applicable.

This is a term-long project and we will use the approach, "***Make it work first, then refactor and make it work with Patterns***".

The goal therefore, is to complete the project in two phases:

Phase 1: Understand the requirements and specification of the project, complete the design and implement it in Java.

You may have to come to speed with many aspects of Java programming: Using inheritance, composition, Swing API, multi-threading, file I/O and so on.

You should have a working prototype in 3 to 4 weeks. During this time, through the lectures and assignments, you will gain an understanding of the concepts of Patterns and applicability of GOF patterns.

It is also possible to use some of the Creational and Structural Design patterns in your implementation at this point.

Phase 2: You will revisit your design and code completed in Phase 1, re-factor the code and re-implement it (and make it better) with design patterns (where applicable) giving reasons for your choices. You will complete the documentation and prepare for the final demo.

Skills learned: This is an application that will test:

- **Object-Oriented Analysis and Design skills,**
- **Knowledge of applying Java language features,**
- **Ability to re-factor code,**
- **Understand and use design patterns effectively**

Introduction

The simulation is of a scenario of events that take place on a distant planet, **Tetra**.

The planet **Tetra** is inhabited by **TetRovers**. TetRovers are generally peaceable people who love to explore and see "what is out there" in the universe.

Therefore, TetRovers value "**StarMaps**" which contain many secrets to the distant galaxies in the universe and were written a long time ago by their ancestors.

Some of TetRovers, called **TetHeroes**, are specially trained to study and protect the StarMaps from falling into evil hands (TetVaders).

TetVaders are TetRovers who have ambitions to take over the universe and have chosen evil ways to do so.

TetVaders are always on the lookout for StarMaps since they will help them to find the worlds out there for them to conquer.

So on Tetra, when our story unfolds, there is just one **TetraVader** who lives in **VaderBase**, a number of (at least two) **TetHeroes** who have their home-bases, a few **TetRovers** (at least two) and a number of **StarMaps** in a number of locations.

Note: A TetRover can be a male or a female; For the sake of convenience, a TetRover/TetHero/TetVader may also be referred to as TRover, THero and TVader.

The description below gives important details of **places** and **actors** in this application.

Tetra Surface (TFace):

To keep things rather simple, in the current version of the project, we will model the **TFace** as a rectangular grid of cells containing the home bases of TVaders and THeros, StarMaps and the TetRovers themselves. Please keep in mind that, in later versions of this project, the StarFace may not always be bounded as a rectangular grid. See a conceptual diagram in file, **Tetra.pdf**.

Each cell in the **TFace** is called a **Location.** A Location may be **empty**, contain a **TetHero base**, a **TetVaderBase**, a **MapBase , a River or a TRover (a TetRover, a TetHero or a TetVader)**.

StarMap: A StarMap consists of information with directions/instructions (text) and/or map(s) (graphic) of distant galaxies.

A StarMap can be either a single packet of information or may be a group of StarMaps (**StarAtlas).** A group of StarMaps is called a **StarAtlas.**

The contents of a StarMap are divided into two parts: a) The Header and b) The Body

The Header contains the **StarMap ID (an unique ID), its location ID, the number of items in the packet**.

In case it is encrypted by a TetHero, the header also contains the **TetHero ID, the date of encryption and the restoration_Counter. The restoration_Counter contains the number of times it was restored (from TetVaderBase) by the specific TetHero.**

The Body of the StarMap contains directions (text) and/or a map (graphic). A StarAtlas consists of a header and a group of StarMaps.

A StarMap (and StarAtlas) can be displayed, encrypted (encoded) and decrypted (decoded).

A StarMap can emit a signal when questioned about its presence.

For example, in OO terms, a message can be sent to a StarMap as shown below;

```
StarSignal signal = StarMap.showSignal(MapId)
```

A StarMap is **encryped (encoded)** in the following manner:

- The TetHero (only a **TetHero** can encrypt a StarMap) adds a layer to the StarMap: The layer consists of the THero ID, date, the text in the map encrytpted and a symbol to be displayed when it is decrypted. If it is a StarAtlas, this is done to every one of the StarMaps in the group.
- The text in the map can be encrypted using a simple strategy (for the current implementation), for example, storing the string reversed).
- Each THero has a unique encryption strategy at any given time, but it is possible to switch the strategy to another one at a different time.

A StarMap is **decrypted (decoded)** in the following manner:

- The THero (only a **THero** who has previously encrypted it, can decrypt a StarMap) displays the StarMap with THero ID, date (of encryption), the text in the map decrytpted with a border showing the symbol, used when encrypted.
- The text in the map can be encrypted using a simple strategy (for the current implementation);
- **Example:**

Assume the original StarMap contained the string, **Text: "Directions to Planet Earth from Tetra**".

After THero (ID: TH12) encrypted it, it looks like: **ID: TH12 Date: April 2012 (Tetra Time) Symbol: '*' Text: "arteT morf htraE tenalP ot sonitceriD".**

When THero decrypts it, it displays:

****

ID: TH12 Date: April 2012 (Tetra Time)

Text: Directions to Planet Earth from Tetra

****

THeroBase: is the home of a THero. There can be only one THeroBase at the most, in any given location. THeroBases are present only in some locations on

TFace.

A THeroBase has an ID that corresponds to the ID of a THero. In our current implementation, a THeroBase can be located only on the edges of the rectangular grid (which represents the TFace).

TVaderBase: is the home of a TVader. Currently, **there can only be one TVaderBase** on Tetra and everyone knows its location. TVaderBase is surrounded by a river on all 4 directions. Therefore, the only way to reach it is by flying using a TFlier.

MapBase: is a place that can hold a StarMap. It can hold only one StarMap at the most and can also be empty (no StarMap).

TFlier: A TFlier is a vehicle that can fly from one location to another on TFace.

TetRovers, TetHeroes and TetVader roam the TFace with different goals.

A **TetRover** moves about TFace, to one **adjacent** location at a time in any randomly chosen direction (North, South, East or West), provided the location is empty (no other TetRover at that location at that time). A TetRover does not have access to a TFlier, therefore cannot fly from one location to the other. A TetRover does not enter a THeroBase or a TVaderBase.

A **TetHero (THero)** moves about TFace, to one **adjacent** location at a time in any randomly chosen direction (North, South, East or West), provided the location is empty (no other TetRover at that location at that time). A TetHero is capable of possessing a **TFlier** (but may not have one at a specific time) and with a TFlier, can fly from his current location to **any other location, in one step**.

A THero can **encrypt** (see the decription above) and **decrypt** a StarMap.

When a THero enters a MapBase, the following events can happen:

1. **If the location is empty**, it only means one thing: TetVader has stolen it. TetHero does the following actions:
    a. TetHero gets his TFlier and flies to TVaderbase.
    b. At TVaderBase, checks if the particular StarMap is there, by sending it a signal. If it is found, makes a clone of it, encrypts the original, stores it back in its base and flies to his homebase with the clone.
        i. If he finds that the StarMap was already encrypted when it was stolen: if it was encrypted by him previously, simply increments the restoration_counter.
        ii. If it was encrypted by another THero, adds his ID to the header.
2. If the MapBase contains a StarMap and is not encrypted, he displays the StarMap. In case of StarAtlas, displays every one of the StarMaps in the group.
3. If the MapBase contains an encrypted StarMap (by him), he decrypts and displays the StarMap.
4. If the MapBase contains an encrypted (not by him) StarMap, adds his ID to the header and displays the **encrypted** contents.

A **TetVader** moves about TFace one **adjacent** location at a time in any randomly chosen direction (North, South, East or West), provided the location is empty (no other TetRover at that location at that time), looking for StarMaps. A TetVader cannot enter a THeroBase. A TetVader also has the ability to fly from one location to **any other location** using a TFlier and is usually travels with one, even when he is moving (one location ata time) instead of flying. When a TetVader enters a location with a StarMap (MapBase), he removes the map from that location and flies to his VaderBase on his TFlier; He, himself, backtracks (retraces the path he followed to come to the current location from his base), one location at a time. TetVader steals all StarMaps, including the encrypted ones.

A Scenario to simulate:

In order to simulate and demonstrate a series of events which take place on Tetra,

you may have to consider the notion of a **TimeStep**. On each Timestep, a number of things (see the description below) take place.

For example, on one timeStep:

- ATetRover/TetHero/TetVader can move a step from one location to another adjacent location.
- A TetHero/TetVader can fly from one location to another location.
- A TetVader can steal a StarMap (empty the MapBase of its contents) and fly it to TVaderBase.
- A TetHero can request to get a TFlier.
- A TetHero can fly from his current location to TVaderBase; he can send a signal to the StarMap.
- A TetHero can retrieve the StarMap from TVaderBase, **encrypt** (see the description above) it and restore it to its base and fly back to his homebase.

Important Tasks:

1. Create and display Tetra surface (TFace)
2. Create actors (TetVaders, TetRovers and TetHeroes, Bases, TFliers, StarMaps …) and populate TFace.
3. Create a Simulation Window with TFace (and all is components) and Graphics User Interface to control the simulation.
4. Create Simulation Steps.
5. Run the simulation using the GUI from step 3.

Note: You are free to enhance the application with interesting details of your choice, as long as they contribute to and enhance the overall theme.