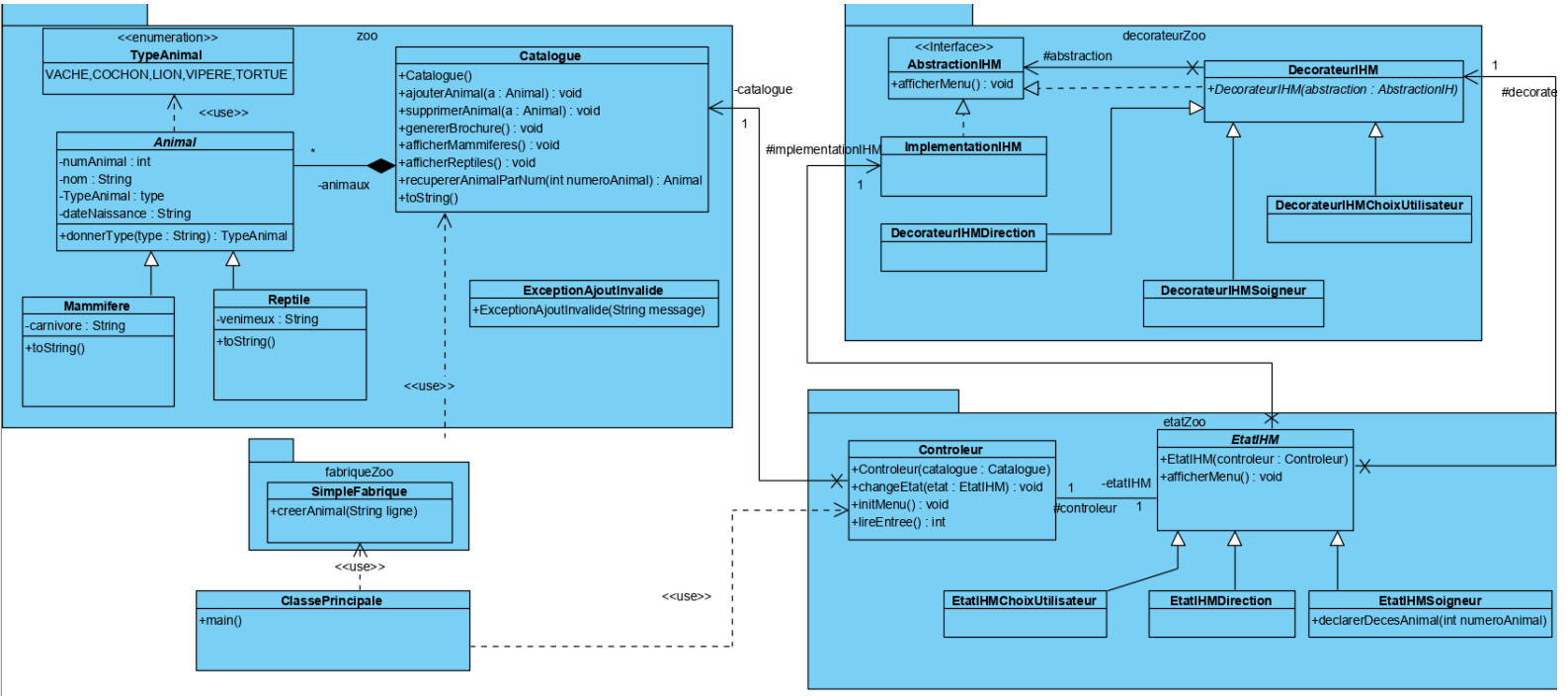
DECOUTY Hugo
TP noté Génie logiciel 2

IDE utilisé : Intellij



Code Java

ClassePrincipale.java

```
import etatZoo.Controleur;
import fabriqueZoo.SimpleFabrique;
import zoo.Catalogue;

import java.io.BufferedReader;
import java.io.FileReader;

public class ClassePrincipale {

    public static void main(String[] args)
    {
        SimpleFabrique simpleFabrique = new SimpleFabrique();
        Catalogue catalogue = new Catalogue();
        BufferedReader lecteurAvecBuffer = null;
        String ligne;

        try
        {
            lecteurAvecBuffer = new BufferedReader(new
FileReader("animaux.txt"));
            while ((ligne = lecteurAvecBuffer.readLine()) != null)
                catalogue.ajouterAnimal(simpleFabrique.creerAnimal(ligne));
            lecteurAvecBuffer.close();
        }
        catch (Exception exc)
```

```
        {
            System.out.println("Erreur à la lecture du fichier de stock" + exc);
        }

        Controleur controleur = new Controleur(catalogue);
        controleur.initMenu();
    }
}
```

Package zoo

Animal.java

```
package zoo;

public abstract class Animal {

    private int numAnimal;
    private String nom;
    private TypeAnimal type;
    private String dateNaissance;

    public static int cptNumAnimal = 1;

    public Animal(String[] tab) throws ExceptionAjoutInvalide {
        this.nom = tab[1];
        this.type = donnerType(tab[2]);
        this.dateNaissance = tab[3];
        this.numAnimal = cptNumAnimal;

        cptNumAnimal++;
    }

    public TypeAnimal donnerType(String type) throws ExceptionAjoutInvalide{
        TypeAnimal typeAnimal = null;
        switch(type){
            case "Vache":
                typeAnimal = TypeAnimal.VACHE;
                break;
            case "Cochon":
                typeAnimal = TypeAnimal.COCHON;
                break;
            case "Lion":
                typeAnimal = TypeAnimal.LION;
                break;
            case "Vipere":
                typeAnimal = TypeAnimal.VIPERE;
                break;
            case "Tortue":
                typeAnimal = TypeAnimal.TORTUE;
                break;
        }
        if(typeAnimal == null){
            throw new ExceptionAjoutInvalide("Ajout impossible de l'animal car
l'animal n'est pas conforme à la spécification");
        } else {
            return typeAnimal;
        }

    }

    public int getNumAnimal() {
        return numAnimal;
```

```
    }

    public String getNom() {
        return nom;
    }

    public TypeAnimal getType() {
        return type;
    }

    public String getDateNaissance() {
        return dateNaissance;
    }

}
```

Catalogue.java

```
package zoo;

import java.util.ArrayList;
import java.util.List;

public class Catalogue {

    private List<Animal> animaux;

    public Catalogue(){
        animaux = new ArrayList<>();
    }

    public List<Animal> getAnimaux() {
        return animaux;
    }

    public void ajouterAnimal(Animal a){
        if(!this.getAnimaux().contains(a)){
            this.animaux.add(a);
        }
    }

    public void supprimerAnimal(Animal a){
        this.animaux.remove(a);
    }

    public void genererBrochure(){
        System.out.println("Génération de la brochure effectuée");
    }

    public void afficherMammiferes(){
        for(Animal a : this.getAnimaux()){
            if(a instanceof Mammifere){
                System.out.println(a.toString());
            }
        }
    }
```

```
    public void afficherReptiles(){
        for(Animal a : this.getAnimaux()){
            if(a instanceof Reptile){
                System.out.println(a.toString());
            }
        }
    }

    public Animal recupererAnimalParNum(int numeroAnimal){
        for(Animal a : this.getAnimaux()){
            if(a.getNumAnimal() == numeroAnimal){
                return a;
            }
        }
        return null;
    }

    public String toString(){
        String s = "";
        for(Animal a : this.getAnimaux()){
            s += a.toString() + "\n";
        }
        return s;
    }
}
```

ExceptionAjoutInvalide.java

```
package zoo;

public class ExceptionAjoutInvalide extends Exception {

    private static final long serialVersionUID = 1L;

    public ExceptionAjoutInvalide(String message)
    {
        super(message);
    }

}
```

Mammifere.java

```
package zoo;


public class Mammifere extends Animal{

    private String carnivore;


    public Mammifere(String[] tab) throws ExceptionAjoutInvalide {
        super(tab);
        if(tab[4].equals("True")){
            this.carnivore = "carnivore";
        } else {
            this.carnivore = "herbivore";
        }
    }

    public String getCarnivore() {
        return carnivore;
    }

    public String toString(){
        return this.getNom() + ", " + this.getType() + ", " +
this.getDateNaissance() + ", " + this.getCarnivore();
    }
}
```

Reptile.java

```
package zoo;

public class Reptile extends Animal {
    private String venimeux;


    public Reptile(String[] tab) throws ExceptionAjoutInvalide {
        super(tab);
        if(tab[4].equals("True")){
            this.venimeux = "venimeux";
        } else {
            this.venimeux = "non venimeux";
        }

    }

    public String getVenimeux() {
        return venimeux;
    }

    public String toString(){
```

```
        return this.getNom() + ", " + this.getType() + ", " +
this.getDateNaissance() + ", " + this.getVenimeux();
    }
}
```

TypeAnimal.java

```
package zoo;

public enum TypeAnimal {
    VACHE,COCHON,LION,VIPERE,TORTUE;
}
```

PACKAGE fabriqueZoo

SimpleFabrique.java

```
package fabriqueZoo;

import zoo.Animal;
import zoo.ExceptionAjoutInvalide;
import zoo.Mammifere;
import zoo.Reptile;

public class SimpleFabrique {

    public Animal creerAnimal(String ligne) throws
ExceptionAjoutInvalide {
        Animal p = null;
        String[] tab = ligne.split(";");
        switch(tab[0])
        {
            case "Mammifere":
                p = new Mammifere(tab);
                break;
            case "Reptile":
                p = new Reptile(tab);
                break;
        }
        return p;
    }

}
```

PACKAGE etatZoo

Controleur.java

```
package etatZoo;

import zoo.Catalogue;

import java.util.Scanner;

public class Controleur {
    private EtatIHM etat;
    private Catalogue catalogue;

    public Controleur(Catalogue catalogue)
    {
        this.catalogue = catalogue;
        this.etat = new EtatIHMChoixUtilisateur(this);
    }

    public Catalogue getCatalogue() {
        return catalogue;
    }

    public void changeEtat(EtatIHM etat)
    {
        this.etat = etat;
        this.etat.afficherMenu(this.catalogue);
    }

    public void initMenu()
    {
        this.etat = new EtatIHMChoixUtilisateur(this);
        this.etat.afficherMenu(this.catalogue);
    }

    public int lireEntree()
    {
        @SuppressWarnings("resource")
        Scanner entree = new Scanner(System.in);
        return entree.nextInt();
    }

}
```

EtatIHM.java

```
package etatZoo;

import decorateurZoo.DecorateurIHM;
import decorateurZoo.ImplementationIHM;
import zoo.Catalogue;

public abstract class EtatIHM {

    protected ImplementationIHM implementationIHM;
    protected DecorateurIHM decorateur;
    protected Controleur controleur;

    public EtatIHM(Controleur controleur)
    {
        this.controleur = controleur;
        this.implementationIHM = new ImplementationIHM();
        this.decorateur = null;
    }

    public abstract void afficherMenu(Catalogue catalogue);


}
```

EtatIHMChoixUtilisateur.java

```
package etatZoo;

import decorateurZoo.DecorateurIHMChoixUtilisateur;
import zoo.Catalogue;

public class EtatIHMChoixUtilisateur extends EtatIHM{

    public EtatIHMChoixUtilisateur(Controleur controleur) {
        super(controleur);
        this.decorateur = new
DecorateurIHMChoixUtilisateur(this.implementationIHM);
    }

    @Override
    public void afficherMenu(Catalogue catalogue) {

        this.decorateur.afficherMenu();

        int choixMenu = this.controleur.lireEntree();
        switch(choixMenu){
            case 1:
                this.controleur.changeEtat(new
EtatIHMDirection(this.controleur));
                break;
            case 2:
```

```
                this.controleur.changeEtat(new
EtatIHMSoigneur(this.controleur));
        }

    }
}
```

EtatIHMDirection.java

```
import zoo.Catalogue;

public class EtatIHMDirection extends EtatIHM{

    public EtatIHMDirection(Controleur controleur) {
        super(controleur);
        this.decorateur = new
DecorateurIHMDirection(this.implementationIHM);
    }

    @Override
    public void afficherMenu(Catalogue catalogue) {

        this.decorateur.afficherMenu();

        int choixMenu = this.controleur.lireEntree();
        switch(choixMenu){
            case 1:
                catalogue.genererBrochure();
                this.afficherMenu(catalogue);
                break;
            case 2:
                System.out.println(catalogue.toString());
                this.afficherMenu(catalogue);
                break;
            case 3:
                catalogue.afficherMammiferes();
                this.afficherMenu(catalogue);
                break;
            case 4:
                catalogue.afficherReptiles();
                this.afficherMenu(catalogue);
                break;
            case 5:
                this.controleur.changeEtat(new
EtatIHMChoixUtilisateur(this.controleur));
                break;
        }


    }
}
```

EtatIHMSoigneur.java

```
package etatZoo;

import decorateurZoo.DecorateurIHMSoigneur;
import zoo.Animal;
import zoo.Catalogue;

public class EtatIHMSoigneur extends EtatIHM {

    public EtatIHMSoigneur(Controleur controleur){
        super(controleur);
        this.decorateur = new
DecorateurIHMSoigneur(this.implementationIHM);
    }

    public void declarerDecesAnimal(int numeroAnimal){
        Animal animal =
this.controleur.getCatalogue().recupererAnimalParNum(numeroAnimal)
;
        if(animal != null){

this.controleur.getCatalogue().supprimerAnimal(animal);
            System.out.println("L'animal " + animal.getNom() + " a
été sauvagement tué !");
        } else {
            System.out.println("Erreur ! Ce numéro d'animal n'est
pas ou plus attribué.");
        }
    }

    @Override
    public void afficherMenu(Catalogue catalogue) {
        this.decorateur.afficherMenu();

        int choixMenu = this.controleur.lireEntree();
        switch(choixMenu){
            case 1:
                System.out.println(catalogue.toString());
                this.afficherMenu(catalogue);
                break;
            case 2:
                System.out.println("----- Qui est mort? -----");

                for(Animal a :
this.controleur.getCatalogue().getAnimaux()){
                    System.out.println("\t" + a.getNumAnimal() + "
- " + a.getNom());
                }
```

```
                int choixAnimalMort =
this.controleur.lireEntree();
                this.declarerDecesAnimal(choixAnimalMort);
                this.afficherMenu(catalogue);
                break;
            case 3:
                this.controleur.changeEtat(new
EtatIHMChoixUtilisateur(this.controleur));
                break;

        }
    }
}
```

package decorateurZoo

AbstractionIHM.java

```
package decorateurZoo;

public interface AbstractionIHM {

    public void afficherMenu();
}
```

DecorateurIHM.java

```
package decorateurZoo;

public abstract class DecorateurIHM implements AbstractionIHM{

    protected AbstractionIHM abstraction;

    public DecorateurIHM(AbstractionIHM abstraction)
{ this.abstraction = abstraction; }


}
```

DecorateurIHMChoixUtilisateur.java

```
package decorateurZoo;

public class DecorateurIHMChoixUtilisateur extends DecorateurIHM {

    public DecorateurIHMChoixUtilisateur(AbstractionIHM
abstraction) {
        super(abstraction);
    }

    @Override
    public void afficherMenu()
    {
```

```
        this.abstraction.afficherMenu();
        System.out.println("1 - Interface direction");
        System.out.println("2 - Interface soigneur");
    }
}
```

DecorateurIHMDirection.java

```
package decorateurZoo;

public class DecorateurIHMDirection extends DecorateurIHM{

    public DecorateurIHMDirection(AbstractionIHM abstraction) {
        super(abstraction);
    }

    @Override
    public void afficherMenu()
    {
        this.abstraction.afficherMenu();
        System.out.println("1 - Générer la brochure");
        System.out.println("2 - Voir la liste des animaux");
        System.out.println("3 - Voir la liste des mammifères");
        System.out.println("4 - Voir la liste des reptiles");
        System.out.println("5 - Retour au menu précédent");
    }
}
```

DecorateurIHMSoigneur.java

```
package decorateurZoo;

public class DecorateurIHMSoigneur extends DecorateurIHM{

    public DecorateurIHMSoigneur(AbstractionIHM abstraction) {
        super(abstraction);
    }

    @Override
    public void afficherMenu()
    {
        this.abstraction.afficherMenu();
        System.out.println("1 - Voir la liste des animaux");
        System.out.println("2 - Déclarer décès");
        System.out.println("3 - Retour au menu précédent");
    }
}
```

ImplementationIHM.java

```
package decorateurZoo;

public class ImplementationIHM implements AbstractionIHM {

    @Override
    public void afficherMenu()
    {
        System.out.println("---------------- Zoo
ALaBonneFranquette -------------");
        System.out.println("0 - Quitter");
    }

}
```

Package test

TestException.java

```
package test;

import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;
import zoo.*;



public class TestException {

    @Test
    public void testPaquerette() throws ExceptionAjoutInvalide {
        String[] tab = {"Mammifere", "Paquerette", "Vache",
"25/12/2015", "False"};
        Animal paquerette = new Mammifere(tab);
        assertEquals(paquerette.getType(), TypeAnimal.VACHE);
    }

    @Test
    public void testVivi() throws ExceptionAjoutInvalide {
        String[] tab = {"Reptile", "Vivi", "Vipere", "25/12/2015",
"True"};
        Animal vivi = new Reptile(tab);
        assertEquals(vivi.getType(), TypeAnimal.VIPERE);
    }

    //Exception doit s'effectuer ici.
    @Test
    public void testChoupi() throws ExceptionAjoutInvalide {
        //Ajout d'un autre animal (un chat ici)
        String[] tabChat = {"Mammifere",
"Choupi","Chat","19/03/2010","False"};
```

```
        Mammifere chat;
        ExceptionAjoutInvalide exception =
assertThrows(ExceptionAjoutInvalide.class, () -> new
Mammifere(tabChat));
        assertEquals("Ajout impossible de l'animal car l'animal
n'est pas conforme à la spécification", exception.getMessage());

    }

}
```