Universidad de las Fuerzas Armadas ESPE

Object Oriented Programming

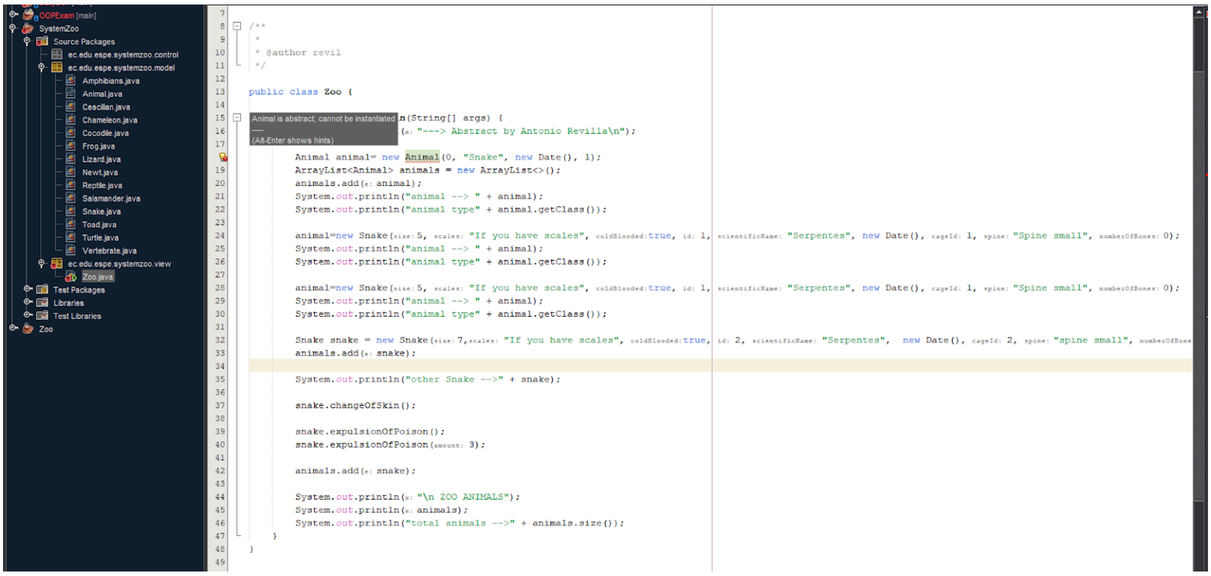
Teacher: Edison Lascano

Student: Revilla Antonio

Nrc: 14575

Homework #15

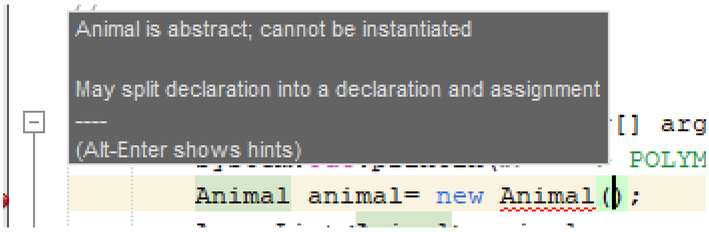
Topic: Abstraction override functions



Animal is abstract; cannot be instantiated

(Alt-Enter shows hints)

```
/**
 *
 * @author revil
 */

public class Zoo {

                                 n(String[] args) {
                                 .(x: "---> Abstract by Antonio Revilla\n");

        Animal animal= new Animal(0, "Snake", new Date(), 1);
        ArrayList<Animal> animals = new ArrayList<>();
        animals.add(e: animal);
        System.out.println("animal --> " + animal);
        System.out.println("animal type" + animal.getClass());

        animal=new Snake(size: 5, scales: "If you have scales", coldBlooded:true, id: 1, scientificName: "Serpentes", new Date(), cageId: 1, spine: "Spine small", numberOfBones: 0);
        System.out.println("animal --> " + animal);
        System.out.println("animal type" + animal.getClass());

        animal=new Snake(size: 5, scales: "If you have scales", coldBlooded:true, id: 1, scientificName: "Serpentes", new Date(), cageId: 1, spine: "Spine small", numberOfBones: 0);
        System.out.println("animal --> " + animal);
        System.out.println("animal type" + animal.getClass());

        Snake snake = new Snake(size: 7,scales: "If you have scales", coldBlooded:true, id: 2, scientificName: "Serpentes",  new Date(), cageId: 2, spine: "spine small", numberOfBone
        animals.add(e: snake);

        System.out.println("other Snake -->" + snake);

        snake.changeOfSkin();

        snake.expulsionOfPoison();
        snake.expulsionOfPoison(amount: 3);

        animals.add(e: snake);

        System.out.println(x: "\n ZOO ANIMALS");
        System.out.println(x: animals);
        System.out.println("total animals -->" + animals.size());
    }
}
```

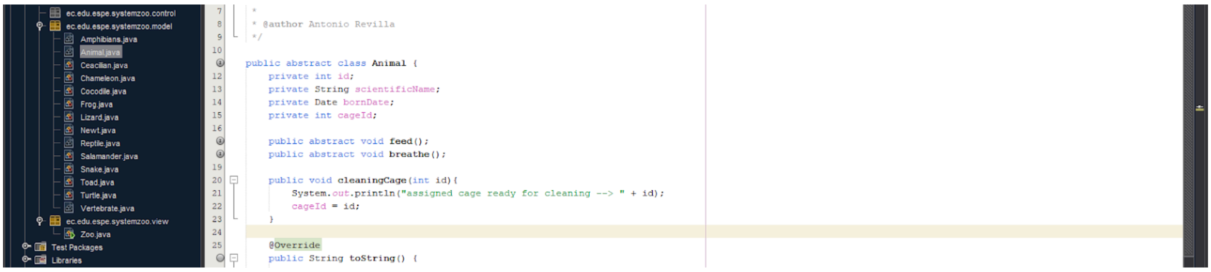


Animal is abstract; cannot be instantiated

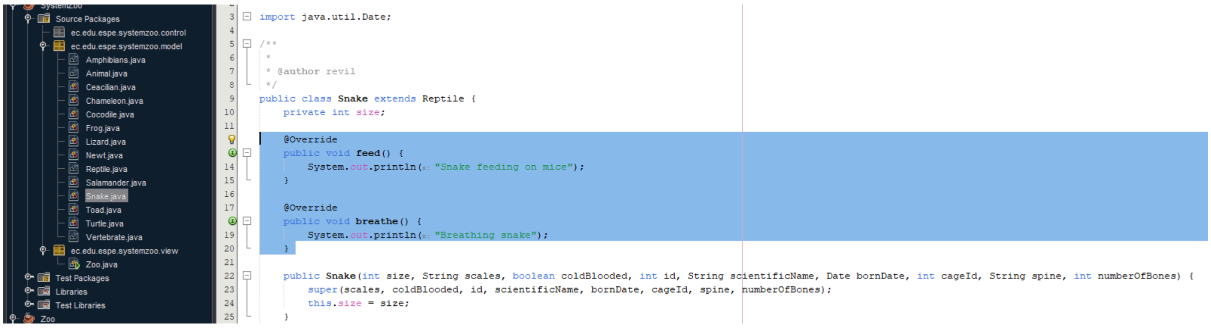
May split declaration into a declaration and assignment

(Alt-Enter shows hints)

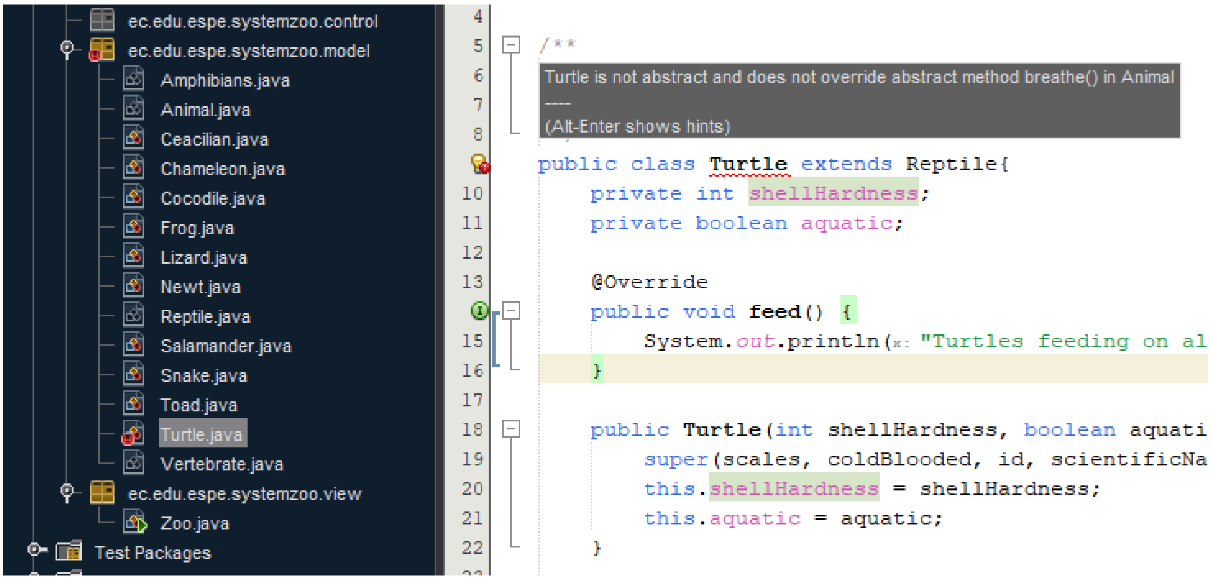
```
[] arg
 POLYM
        Animal animal= new Animal();
```



```
 *
 * @author Antonio Revilla
 */

public abstract class Animal {
    private int id;
    private String scientificName;
    private Date bornDate;
    private int cageId;

    public abstract void feed();
    public abstract void breathe();

    public void cleaningCage(int id){
        System.out.println("assigned cage ready for cleaning --> " + id);
        cageId = id;
    }

    @Override
    public String toString() {
```

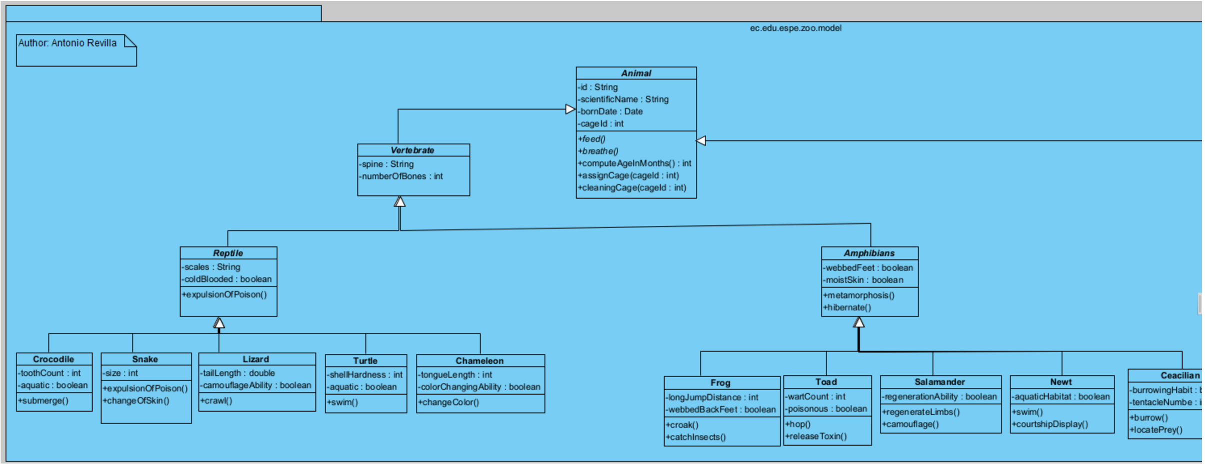


```
import java.util.Date;

/**
 *
 * @author revil
 */
public class Snake extends Reptile {
    private int size;

    @Override
    public void feed() {
        System.out.println(x: "Snake feeding on mice");
    }

    @Override
    public void breathe() {
        System.out.println(x: "Breathing snake");
    }

    public Snake(int size, String scales, boolean coldBlooded, int id, String scientificName, Date bornDate, int cageId, String spine, int numberOfBones) {
        super(scales, coldBlooded, id, scientificName, bornDate, cageId, spine, numberOfBones);
        this.size = size;
    }
```



Turtle is not abstract and does not override abstract method breathe() in Animal
----
(Alt-Enter shows hints)

```
/**

public class Turtle extends Reptile{
    private int shellHardness;
    private boolean aquatic;

    @Override
    public void feed() {
        System.out.println(x: "Turtles feeding on al
    }

    public Turtle(int shellHardness, boolean aquati
        super(scales, coldBlooded, id, scientificNa
        this.shellHardness = shellHardness;
        this.aquatic = aquatic;
    }
```


ec.edu.espe.zoo.model
Author: Antonio Revilla
Animal
-id : String
-scientificName : String
-bornDate : Date
-cageId : int
+feed()
+breathe()
+computeAgeInMonths() : int
+assignCage(cageId : int)
+cleaningCage(cageId : int)
Vertebrate
-spine : String
-numberOfBones : int
Reptile
-scales : String
-coldBlooded : boolean
+expulsionOfPoison()
Amphibians
-webbedFeet : boolean
-moistSkin : boolean
+metamorphosis()
+hibernate()
Crocodile
-toothCount : int
-aquatic : boolean
+submerge()
Snake
-size : int
+expulsionOfPoison()
+changeOfSkin()
Lizard
-tailLength : double
-camouflageAbility : boolean
+crawl()
Turtle
-shellHardness : int
-aquatic : boolean
+swim()
Chameleon
-tongueLength : int
-colorChangingAbility : boolean
+changeColor()
Frog
-longJumpDistance : int
-webbedBackFeet : boolean
+croak()
+catchInsects()
Toad
-wartCount : int
-poisonous : boolean
+hop()
+releaseToxin()
Salamander
-regenerationAbility : boolean
+regenerateLimbs()
+camouflage()
Newt
-aquaticHabitat : boolean
+swim()
+courtshipDisplay()
Ceacilian
-burrowingHabit : b
-tentacleNumbe : i
+burrow()
+locatePrey()