Ollscoil na Gaillimhe
University of Galway

# CT2106
## Object Oriented Programming

**Dr. Frank Glavin**
Room 404, IT Building
Frank.Glavin@University*of*Galway.ie
School of Computer Science

University *of* Galway.ie

# Instructions from last week

## Food:

Make Food an abstract class
Give it two abstract methods *getCalories* and *getFat* with a return type *int*

## Animal: make *eat* method abstract

- Create an abstract subclass of Food called **Vegetable**
- Create a concrete subclass of Vegetable called **Seed**
- Seed has two fields *calories* and *fat*
- Canary must implement a concrete version of the *eat* method
- Canary's eat method checks if Food object is an *instanceof* Seed; if it is, the Canary calls Food's getCalories method and moves the distance returns. She also calls the sing method.

Ollscoil na Gaillimhe
University of Galway

# Slight revision to these instructions

- We'll drop the *getFat* method from Food – as I don't plan to use it

- Canary's *eat* method  should do the following:
  - Check if the Food object is null
  - Checks if Food object is an *instanceof* Seed;
  - if it is a Seed, the canary calls Food's *extractEnergy* method ~~and moves the distance returns~~ and adds the value returned to its own energy level
  - It also calls the sing method (because it is now well fed)

Ollscoil na Gaillimhe
University of Galway

# This lecture

- We'll look at some modelling issues
- We'll introduce the background for the next topic: **interfaces**

- To introduce this topic we'll model a **food chain**

Ollscoil na Gaillimhe
University of Galway

# Food Chain

Download the zip file provided in the Week 8 folder

Create a new Project in BlueJ

In the Workbench menu, select

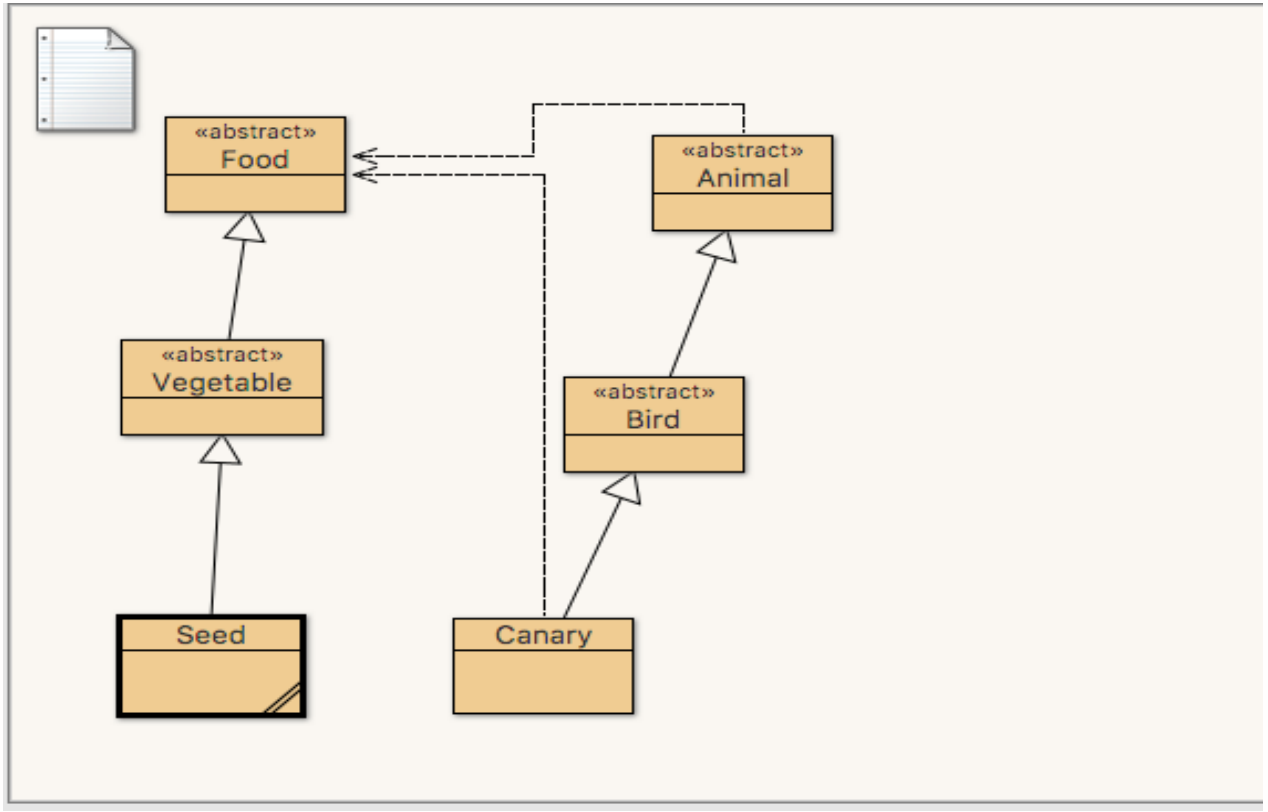
Project -> Open Zip/Jar

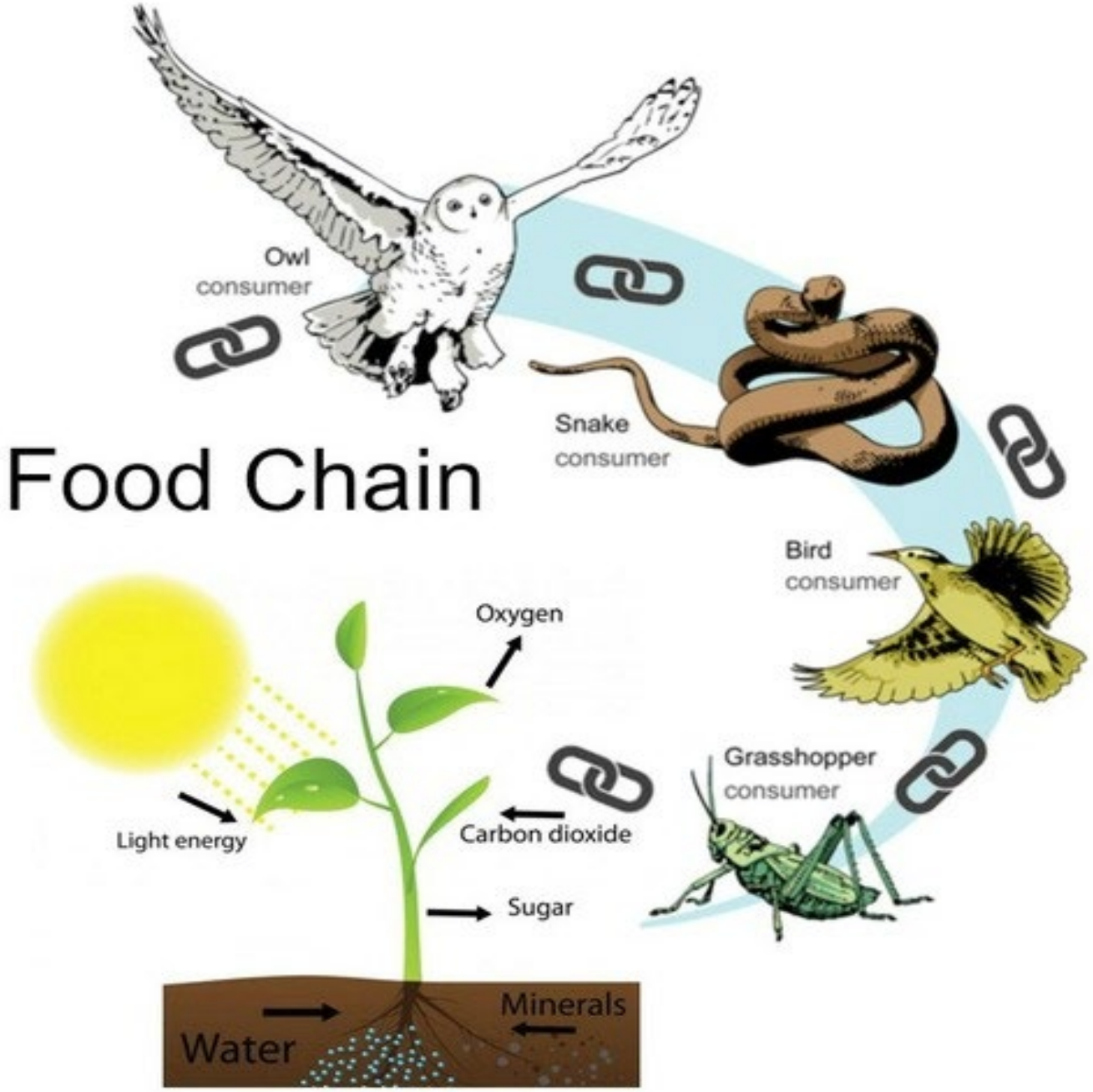
Then compile the Project

Ollscoil na Gaillimhe
University of Galway

# Blue J workbench

Rearrange the class icons to give you something like






Ollscoil na Gaillimhe
University of Galway

# Our Food Chain

Seeds

Canaries

Cats

- Canaries eat Seed
- Cats eat Canaries
- Energy passes from Seeds to the Canary to the Cat

Ollscoil na Gaillimhe
University of Galway

# Canaries eat Seed

- Animal class has an abstract eat method
- Canary has to *override* the eat method it has inherited from Animal
- We now have to write the specific code to allow Canaries eat Seed

```
@Override
public void eat(Food food){
    // TODO

}
```

**Note how the eat method takes as input a Food reference**

Ollscoil na Gaillimhe
University of Galway

# Canary's eat method

- Canary's *eat* method  should do the following:
  1. Check if the Food object is null
  2. Checks if Food object is an *instanceof* Seed;
  3. If it is a Seed, the canary calls the *extractEnergy* method and *adds* the value returned to its own energy level
  4. It also calls the sing method (because it is now well fed)
- I would also suggest that this method is modified to return a boolean depending on whether the Food is edible (e.g it is a Seed or not)

Ollscoil na Gaillimhe
University of Galway

# First: Animal energy

As an Animal object  gets energy from the Food objects it can consume, it needs a numeric field *energy* to hold this value

*This field can then be inherited by all Animal objects, including Canary*

```
public abstract class Animal
{
    // instance variables - replace the example
    boolean hasSkin;
    boolean breathes;
    String colour;
    int energy;
```

Ollscoil na Gaillimhe
University of Galway

# getEnergy

You will also need an accessor (getter) method for the new energy field in Animal

```
/**
 * getter method for energy field
 * All subclasses inherit this method
 */
public int getEnergy(){
    return energy;
}
```

Please remember Getter/Setter methods are not optional.

**You must use them to access the fields of an object**

Ollscoil na Gaillimhe
University of Galway

# extractEnergy

An abstract method defined in the Food class

**It must be implemented** in one of the subclasses of Food

We implement it in the Seed Class. **Implement this method, as described**

```
/**
 * returns the current value for Calories
 * and then sets the calory value to zero
 * i.e. the energy has been extracted from Seed
 */
@Override
public int extractEnergy(){
    //TODO
    return 0;
}
```

Ollscoil na Gaillimhe
University of Galway

# All Food has calories

I originally declared the *calories* field in the Seed class
But *all* Food has calories
Therefore, we should remove the *calories* declaration in Seed and move it to the Food class

```
public abstract class Food
{
    // instance variables - replace
    int calories;
```

It can be then inherited by all sub-classes of Food, including Seed

Ollscoil na Gaillimhe
University of Galway

# Implement Canary's eat method

Canary's *eat* method  should do the following:

1. Check if the Food object is null
2. Checks if Food object is an *instanceof* Seed;
3. If it is a Seed, the canary calls the *extractEnergy* method and *adds* the value returned to its own energy level
4. It also calls the sing method (because it is now well fed)

I would also suggest that this method is modified to return a boolean depending on whether the Food is edible (e.g it is a Seed or not)

Ollscoil na Gaillimhe
University of Galway

# Test first part of the food chain

Seeds

Canaries

- Each seed has 10 calories
- If a Canary eats 3 seeds, its energy level should be 30

# In Code Pad

Or in a main method, type the following

```
Seed millet = new Seed();
Seed sunflower = new Seed();
Seed hayseed = new Seed();
Canary bluey = new Canary("Bluey");
bluey.eat(millet);
bluey.eat(sunflower);
bluey.eat(hayseed);
System.out.println(bluey.getEnergy());
```

This should print out the value 30

Ollscoil na Gaillimhe
University of Galway

# Part 2 of our food chain



Seeds

Canaries

Cats

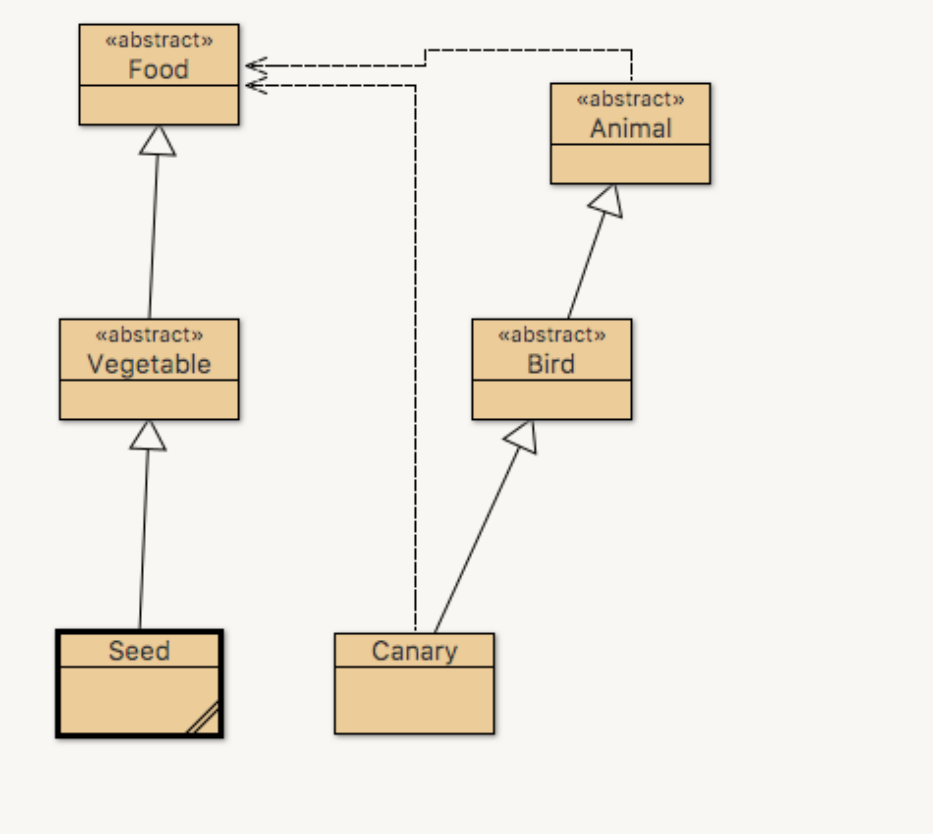
- Cats eat Canaries
- Energy passes from the canary to the Cat

# Part 2

Currently the class structure looks like this

You are now going to add two more classes

- Feline (abstract)
- Cat (concrete)




Ollscoil na Gaillimhe
University of Galway

Feline class (abstract)

**Extends Animal**

**Fields**

hasFur

**Overrides**

move() method

Cat class (concrete)

**Extends Feline**

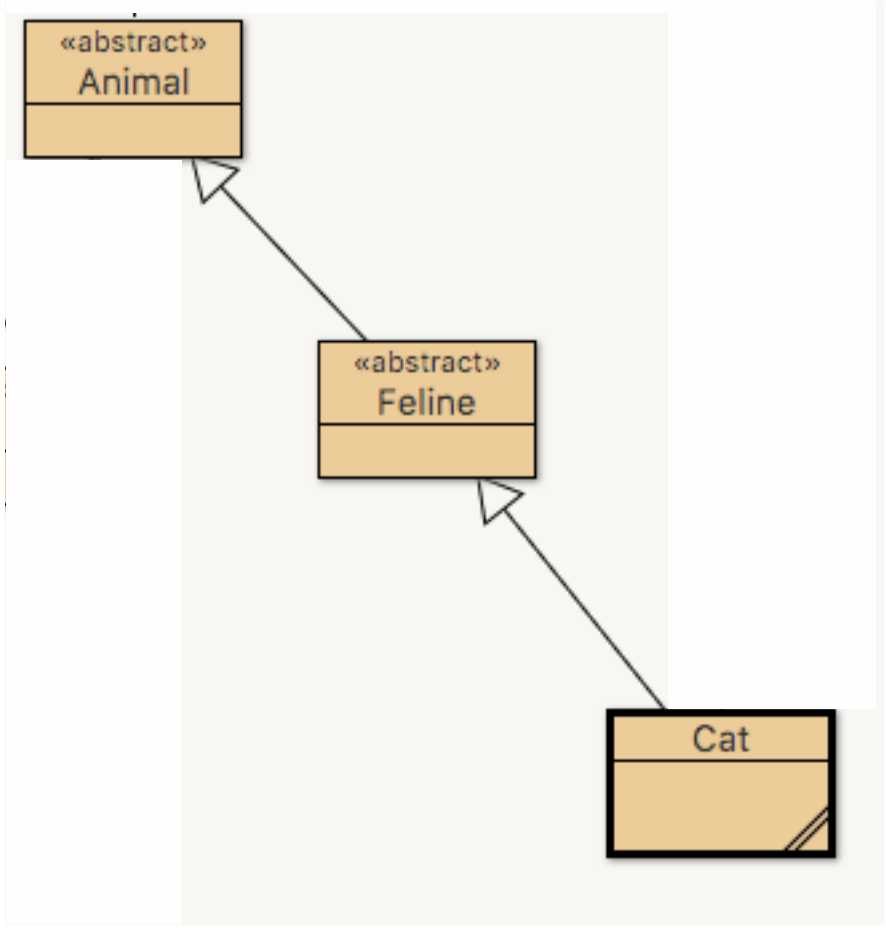
**Fields**

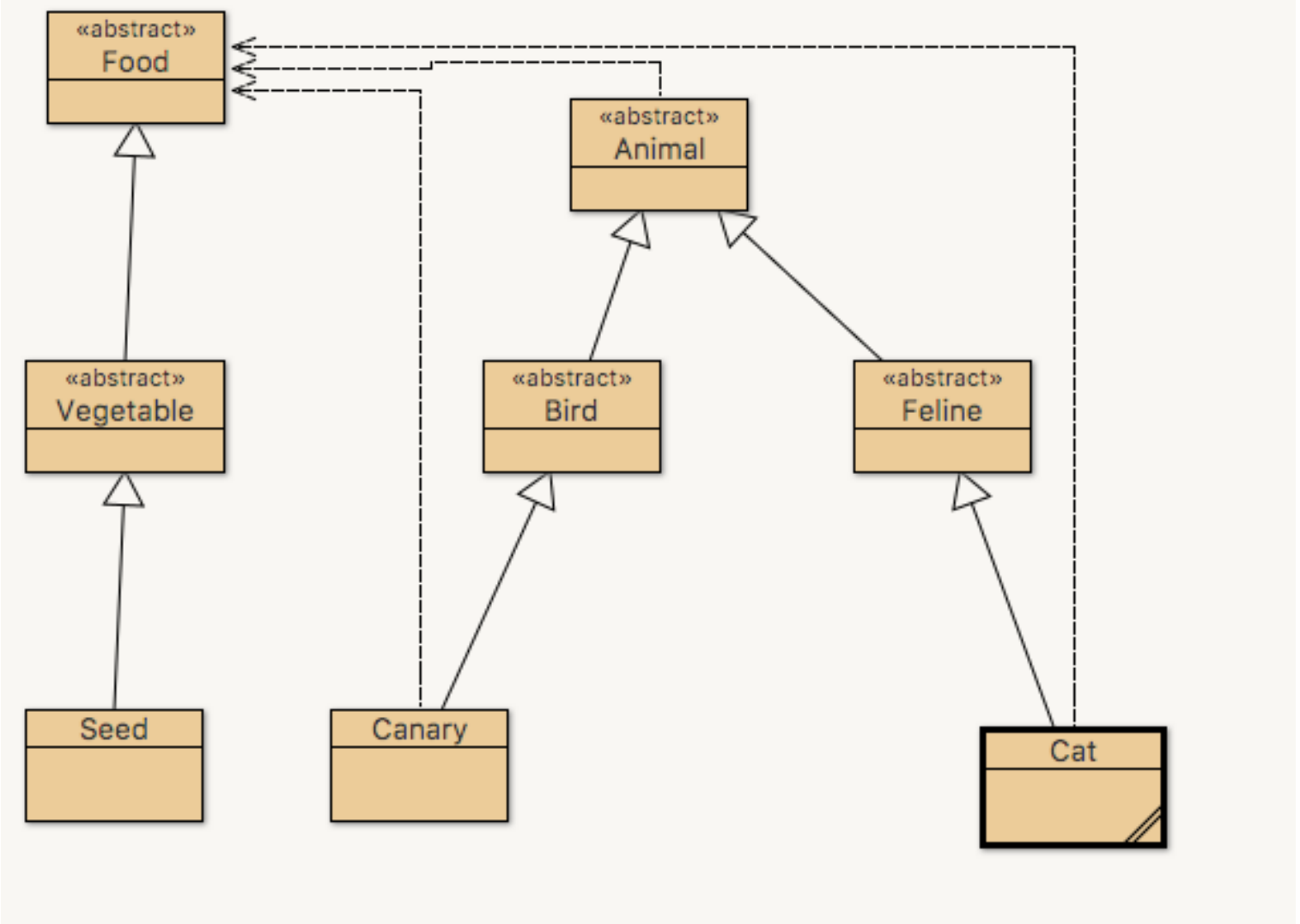
name

**Overrides**

colour field (colour=black)

eat (Food) method




«abstract»
Food
«abstract»
Animal
«abstract»
Vegetable
«abstract»
Bird
«abstract»
Feline
Seed
Canary
Cat