E

Enums and Switch Expressions

Project ID: 6124

Added solution
Jonas Kaad authored 1 month ago

Forked from an inaccessible project.

Name	Last commit	Last update
SOLUTION	Added solution	1 month ago
src/main/java/vop	Task reset	2 months ago
.gitignore	Initial push	2 months ago
pom.xml	Initial push	2 months ago
readme.md	Task reset	2 months ago

readme.md

Enums and Switch Expressions

This task is created to make you a bit more comfortable using enums and introduce you to switch expressions.

You may already have used the `switch` statement, however since JDK 12, Java allows you to use it as an expression as well - neat!

For instance, let's say we want to map a creature's type, ordinarily we would do something like this:

```
String creature = "person";
String type = "";
switch (creature)
{
    case "person":
        type = "humanoid";
        break;
    case "chihuahua":
    case "bulldog":
        type = "canine";
        break;
    case "sword fish":
        type = "fish";
        break;
    default:
        throw new Exception("Invalid creature");
}
```

Using switch expressions we can rewrite it:

```
String creature = "person";
String type = switch (creature) {
    case "person" -> "humanoid";
```

```
    case "chihuahua", "bulldog" -> "canine";
    case "sword fish" -> "fish";
    default -> throw new Exception("Invalid creature");
};
```

The task

In this task we'll be working with Bats. `bats.Bats.all()` provides a list of bats with the following attributes:

- `name : String` - name of the bat
- `populationStatus : String` - The status of its population:
  - Increasing
  - Declining
  - Stable
  - Unknown
- `feedingGroups : String` - What food the bat eats:
  - Frugivore,
  - Insectivore,
  - Omnivore,
  - Sanguivore,
  - Nectarivore,
  - Carnivore

However, there is a problem; for some reason the original developer thought it would be a good idea to save `populationStatus` and `feedingGroups` as strings 😱.

As such, you are tasked with converting each bat in the list to an instance of `BatWithEnum` where the `populationStatus` and `feedingGroups` attributes have been switched out with appropriate enums.

Task 1 - BatStatistics - convert

The `convert` method needs to be implemented.

- Create a new `List<BatWithEnum>` local variable that can hold the new `BatWithEnum` instances you will be creating.
- Iterate over the parameter/list `bats` (the parameter referenced gets instantiated in the main method) and create a new instance of `BatWithEnum` for each of them
  - when creating a new `BatWithEnum` instance:
    - use a switch expression to set the `populationStatus` constructor parameter depending on which of the ENUMs correlate to the `populationStatus` of the current bats object
    - For the default case THROW an `InvalidPopulationStatusException` that is propagated to the calling method (Do not handle the exception within the `convert` method).
    - use `FeedingGroups.valueOf(..)` to set the `feedingGroups` constructor parameter
- return the list from the method

If implemented correctly, you should be able to run the `main` method of the `BatStatistics` -class without getting any errors.

Task 2 - BatStatistics - getPopulationStatusMap and getFeedingGroupsMap

Now that we have converted our `List<Bat>` to `List<BatWithEnum>` , we need to map them, such that it's easier to look up bats with a given `PopulationStatus` and `FeedingGroups` .

This will allow us to retrieve bats that are `Insectivore` and get a complete list of bats that fall under that category for example.

The procedure for `getPopulationStatusMap` and `getFeedingGroupsMap` are the same other the fact that you will be using different enums.

Implement `getPopulationStatusMap` :

- Create a local instance of `Map<PopulationStatus, List<BatWithEnum>>` that holds the bats.
- Iterate over the list of bats in the `bats` parameter.
- For each bat, find the list of bats of the same type (hint: you can use `map.get(bat.getPopulationStatus())`) and save it to a local variable within your loop
  - The returned value from `map.get(bat.getPopulationStatus())` is going to be null during the first iteration of each type, as the list doesn't exist within the map. If that's the case you should create a new `ArrayList` for the bat Type (Hint: `map.put(<type>, new ArrayList<>())`)
  - Add the `BatWithEnum` instance to the Arraylist
  - Continue until you have mapped all bats

- return the map

Implement `getFeedingGroupsMap` :

- Create a local instance of `Map<FeedingGroups, List<BatWithEnum>>` that holds the bats.
- Iterate over the list of bats in the `bats` parameter.
- For each bat, find the list of bats of the same type (hint: you can use `map.get(bat.getFeedingGroups())`) and save it to a local variable within your loop
  - The returned value from `map.get(bat.getFeedingGroups())` is going to be null during the first iteration of each type, as the list doesn't exist within the map. If that's the case you should create a new `ArrayList` for the bat Type (Hint: `map.put(<type>, new ArrayList<>())`)
  - Add the `BatWithEnum` instance to the Arraylist
  - Continue until you have mapped all bats
- return the map

To test your implementation uncomment the remaining part of the `main` method and run it.

If done correctly, you should get an output that starts like this:

```
The following bats are carnivorous:
 - Lyroderma lyra
 - Megaderma spasma
 - Mimon bennettii
 - Noctilio albiventris
 - Noctilio leporinus
 - Vampyrumspectrum

The following bats are declining in population:
 - Anoura cultrata
 - Aproteles bulmerae
 - Artibeus fraterculus
 - Austronomus australis
 - Balantiopteryx infusca
 - Balantiopteryx io
 - Barbastella barbastellus
 - Boneia bidens
 - Chaerephon bregullae
 - Chalinolobus dwyeri
 - Chalinolobus tuberculatus
 - Chilonatalus micropus
 - Chilonatalus tumidifrons
 - Coelops robinsoni
 - Coleura seychellensis
 - Corynorhinus mexicanus
 - Craseonycteris thonglongyai
 - Dobsonia chapmani
 - Dobsonia emersa
 - Dyacopterus rickarti
 - Dyacopterus spadiceus
...
```

```java
package vop;


import vop.bats.*;
import vop.exceptions.
InvalidPopulationStatusException;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class BatStatistics {

    public static List<BatWithEnum> convert(List<Bat
> bats) throws InvalidPopulationStatusException {
        List<BatWithEnum> batWithEnumList = new
ArrayList<>();

        for(Bat bat : bats){
            BatWithEnum myBat;

            String populationStatusString = bat.
getPopulationStatus();
            PopulationStatus populationStatus =
switch (populationStatusString) {
                case "Unknown" -> PopulationStatus.
Unknown;
                case "Stable"  -> PopulationStatus.
Stable;
                case "Increasing" -> PopulationStatus
.Increasing;
                case "Decreasing" -> PopulationStatus
.Decreasing;
                default -> throw new
InvalidPopulationStatusException();
            };
            myBat = new BatWithEnum(bat.getName(),
populationStatus, FeedingGroups.valueOf(bat.
getFeedingGroups()));
            batWithEnumList.add(myBat);
```

```
30          }
31
32          return batWithEnumList;
33      }
34
35      public static Map<PopulationStatus, List<
    BatWithEnum>> getPopulationStatusMap(List<BatWithEnum
    > bats)
36      {
37          Map<PopulationStatus, List<BatWithEnum>>
    populationStatusListMap = new HashMap<>();
38
39          for(BatWithEnum bat : bats){
40              List<BatWithEnum> popStat =
    populationStatusListMap.get(bat.getPopulationStatus
    ());
41              if(popStat == null){
42                  populationStatusListMap.put(bat.
    getPopulationStatus(), new ArrayList<>());
43                  populationStatusListMap.get(bat.
    getPopulationStatus()).add(bat);
44              }
45              else {
46                  popStat.add(bat);
47              }
48          }
49
50          return populationStatusListMap;
51      }
52
53      public static Map<FeedingGroups, List<BatWithEnum
    >> getFeedingGroupsMap(List<BatWithEnum> bats)
54      {
55          Map<FeedingGroups, List<BatWithEnum>>
    feedingGroupsListMap = new HashMap<>();
56
57          for(BatWithEnum bat : bats){
58              List<BatWithEnum> feedGroup =
    feedingGroupsListMap.get(bat.getFeedingGroups());
59              if(feedGroup == null){
60                  feedingGroupsListMap.put(bat.
```

```
60 getFeedingGroups(), new ArrayList<>());
61                 feedingGroupsListMap.get(bat.
   getFeedingGroups()).add(bat);
62             }
63             else {
64                 feedGroup.add(bat);
65             }
66         }
67 
68         return feedingGroupsListMap;
69     }
70 
71     public static void main(String[] args) {
72 
73         try {
74             List<Bat> bats = Bats.all();
75             List<BatWithEnum> convertedBats =
   convert(bats);
76 
77             System.out.println("The following bats
   are carnivorous:");
78             BatStatistics.getFeedingGroupsMap(
   convertedBats)
79                     .get(FeedingGroups.Carnivore)
80                     .forEach(x -> System.out.println
   (" - " + x.toString()));
81             System.out.print("\n");
82             System.out.println("The following bats
   are sanguines:");
83             BatStatistics.getFeedingGroupsMap(
   convertedBats)
84                     .get(FeedingGroups.Sanguivore)
85                     .forEach(x -> System.out.println
   (" - " + x.toString()));
86             System.out.print("\n");
87             System.out.println("The following bats
   are declining in population:");
88             BatStatistics.getPopulationStatusMap(
   convertedBats)
89                     .get(PopulationStatus.Decreasing
   )
```

```
90                      .forEach(x -> System.out.println
   (" - " + x.toString()));
91
92          } catch (InvalidPopulationStatusException e
   ) {
93              e.printStackTrace();
94          }
95      }
96
97 }
98
```