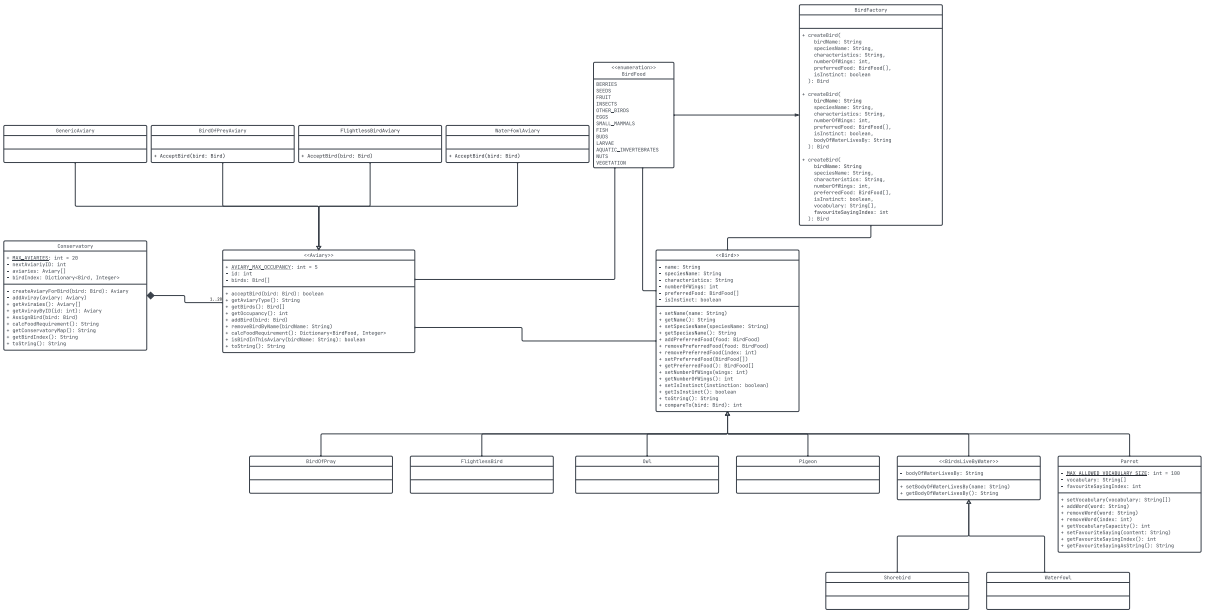
Lab 01. Birds

Design UML



Design Notes

- As how the observation data of birds is not specified in the requirements, I assume the observation data is a string that documents the defining characteristics of the bird.
- The food choice of the bird is limited to a defined set of foods, so I created a `enum BirdFood` to restrict the possible food choices.

Approach and Testing Plans

Class `BirdFactory`

This class use `createBird()` with function overriding to:

- Based on the `characteristics` passed-in, determine the class of the bird
- Call appropriate constructor depending on the class of bird

Testing Plan

- Passing in parameters of each non-abstract subclass of `Bird`
    - **Expected:** Given appropriate parameters, the correct class constructor is called.
- Passing in parameters that has undocumented characteristics, or characteristics does not match any defined class of birds, or unreasonable/incomplete parameters (i.e. missing or too much food choice, negative/extremely large number of wings, non-water birds passed in body of water lives by, non-parrots passed in the vocabulary and favorite saying, etc.)
    - **Expected**: Throws an `IllegalArgumentError` with a comment on the detailed reason why the appropriate constructor cannot be called.

Class `Bird`

This abstract base class represents a bird.

Testing Plan

As this class is not instantiable, the test will be using one of its child classes.

- `getter` and `setter` functions will be test in-pair to ensure the setting and getting the private attributes are working properly.
    - **Expected**: After using `setter` function to set an attribute, its respective `getter` function returns the same value that was set.
        - For `setPreferredFood()`, an `IllegalStateException` is thrown if param `BirdFood` has a length not in `2..4` .
        - For `setNumberOfWings()`, an `IllegalStateException` is thrown if param `wings` is negative.
- `addPreferredFood()` - with the `preferredFood.length < 4`
    - **Expected:** The passed-in food is added at the end of `preferredFood` list.
- `addPreferredFood()` - with the `preferredFood.length == 4`
    - **Expected:** A `IllegalStateException` is thrown explaining that the list is full.
- `addPreferredFood()` - with the food passed in already in the list.
    - **Expected:** The `preferredFood` list does not change. A message is printed as a soft warning.

- `removePreferredFood()` - with the index or food present in the list
  - **Expected:** The selected food is removed from the `preferredFood` list.
- `removePreferredFood()` - with the index or food not present in the list
  - **Expected:** The `preferredFood` list does not change. A message is printed as a soft warning.

Class `BirdsLiveByWater`

This class represents birds that live by a body of water, a new attribute is defined for this class and we only additionally implemented the respective getter and setters.

Class `Parrot`

This class represent parrots.

Testing Plan

- Aside from getter and setters, the only thing that needs to be tested is `setVocabularyCapacity()` , which according to the specification, param should be ranging in `1..100` . Plus regular test for getter and setter for this attribute, we also test for passing in illegal arguments.
  - **Expected:** Throws an `IllegalArgumentError` if param `capacity` is not in `1..100`.