Project 8

PROJECT 8

For this project, we were given freedom to choose our own path. The only basic instructions of the project were to implement a game of our choosing. The only rule was that we had to use a data structure that we have used or studied in class. We could make a game or something interesting using arrays, LinkedLists, BinaryTrees, etc, etc.. For my project, I decided to try and use something we haven't actually implemented fully before -- Binary Trees.

== Project Plan ==

My plan and idea for this project was to make a Ornithology Lab Bird Identification game using a BinaryTree. I am a really big bird enthusiasts and I use an application on my iPhone called Merlin Bird ID to help me identify birds while I am out birding in the field. The app basically takes you through a number of two answer questions, sort of like a Binary Tree would. So that is where I got my inspiration for this project. I wanted to make an interactive project that had buttons and text displays. With the help of the basic.java document provided to us, StackOverflow, and CP Majgaard, I was able to implement JFrame knowledge with JButtons, JFrame images, and other JFrame and GUI things.

== Project Design ==

The design for this project included the implementation of a total of three java classes. 1) BinaryTree Node class to manage the nodes of my BirdID session, 2) a BirdID class to manage to methods involved in the playing of the session and 3) an Identify class that actually set up the session and ran the ID.

BTNode.java

Fields: BTNode A, B and Strings aChoice, bChoice, and instructions

There were a bunch of accessor and mutator methods in this class. For example, setA(), getB(), setAChoice(), getBChoice(), setInstructions(), etc.

isInstructions() : (boolean) this method was important and I actually implemented this later on in the coding. This was useful for helping me determine where I am on the tree in relation to the leaves, or the end of the tree. I the BTNodes for both A and B were null, then this method returned false, indicating that the tree had reached its leaf.

BirdID.java

Fields: BTNode root and current --to help keep track throughout the tree

restartID(): this method basically was used to restart the ID session

isLeaf(): this method used isInstruction to check is we have reached the end of the tree.

The next two methods in this class were very important.

getA(): (String): this method returns the instructions that result from choosing the aChoice button option. If you choose aChoice, then this method will navigate you to the next set of instructions in the tree. BUT, if aChoice returns null instructions then you have reached the leaf and then we return a String message including the list of possible birds that it could have been. There is an identical getB() method too.

```
//returns the instructions of the aChoice button
//if you choose aChoice, this will navigate to the next set of instructions
public String getA(){

    String message = this.current.getA().getInstruction();

    //if the current aChoice returns null instructions then you have reached the leaf
    if(!this.current.getA().isInstruction()){
        current = current.getA();
        //return the list of birds you might have seen
        //this is the message of the final BTNode
        return "<html>You saw one of these birds: " + message + ".</html>";
    }
    current = current.getA();
    return message;
}
```

Identify.java

Fields: there are a lot of JFrame fields here that I got help with from CP and the basic.java file. Also, you got a BirdID session field.

The constructor method is very important here. It includes the set-up for all the BTNodes. This was very difficult to code and keep track of the parentheses because of all the inner-stacking BTNodes.

Also, there are a bunch of JFrame implementations in this class that set up buttons, text, and images. I received guidance here as well.

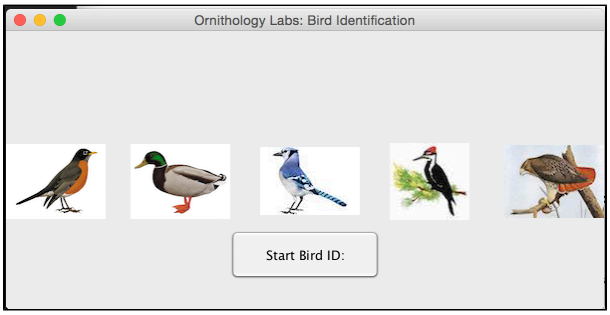
toggleChoiceMode(): this is a method I was told I needed to help me toggle which buttons are shown or not shown using a boolean value.

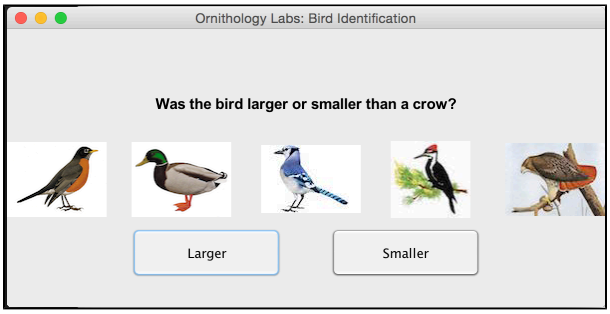
actionPerformed(): this is basically the main method because it manages what to do in the session when each button is pushed. It covers every possible route throughout the tree and also takes into consideration reaching the end of the tree!

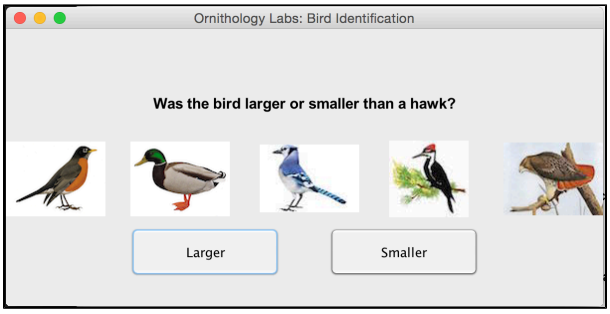
== Solution ==

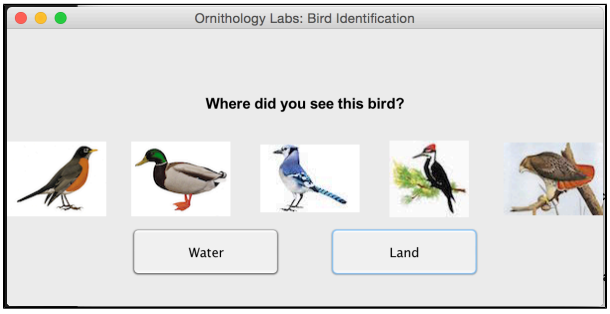
Overall, implementing a BinaryTree with adjustable/edit-able nodes and working with JFrame, which is something I have never seen before, were the two most difficult pieces of this project. At first, the buttons wouldn't let me have multiple choices for the Children-Node answers, but then I edited the parameters of the BTNode to take in the aChoice and bChoice strings that were used as the JButtons as well as the children nodes of the tree. Also, identifying the end of the tree and printing out the list of the birds was an interesting task.

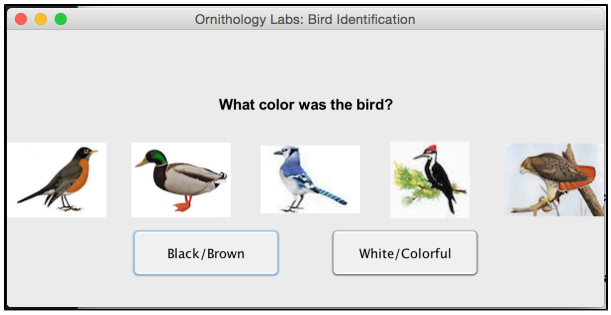
Here is a collection of screenshots of my project:

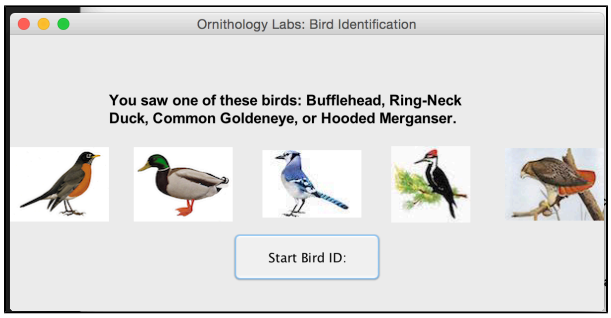












Extension 1. I completed this extension during my project. Instead of just simply implementing a few buttons with two choices for answers, I was able to make my code and my BTNodes to take a "Choice" parameter that gave me the ability to have more than just to possible answers like a simple "Yes" or "No" game. I was able to add parameters throughout my tree to update and change the two answer choices.

Extension 2. For this extension, I made the JFrame more visually appealing and researched how to add images to my JFrame. I added five png's of different common New England birds to my Bird Identification game! Here is a snippet of code on how I added images to my JFrame:

```
//extension
//found JFrame image capabilities on StackOverflow
//decorated the JFrame with an assortment of bird pictures
JLabel imageLabel = new JLabel( new ImageIcon("robin.png"));
add(imageLabel);
imageLabel.setBounds(-150, -50, 400, 400);
imageLabel.setVisible(true);

JLabel imageLabel1 = new JLabel( new ImageIcon("mallard.png"));
add(imageLabel1);
imageLabel1.setBounds(-25, -50, 400, 400);
imageLabel1.setVisible(true);

JLabel imageLabel2 = new JLabel( new ImageIcon("bluejay.png"));
add(imageLabel2);
imageLabel2.setBounds(105, -50, 400, 400);
imageLabel2.setVisible(true);
```

What I learned. I learned ALOT of new things in this project. First off, I became comfortable with implementing and using BinaryTrees and TreeNodes. Also, I learned a whole new part of java with the JFrame and GUI knowledge.

Who helped me. I worked alongside CS231 classmates Steven Parrott and Brendan Doyle.* *I received help and guidance from CP Majgaard. And I used a lot of the Google machine.