C#   
ABSTRACT CLASSES

## Objective

The primary objective of this lab is to provide you with the skills necessary to be able to:

* Create and work with abstract classes
* Create classes in a class library type of project

# Part 1

In the previous lab, it does not make sense to create an object of type Shape.

Please mark the Shape class as abstract and then run your code.

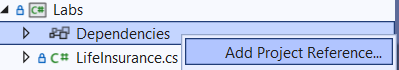
# Part 2

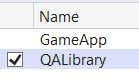
You are going to put Penguins, Ducks and Fish in a collection.

However, you will create these classes in a **Class library** type of project.   
As the name implies, this type of project holds classes and compiles the code to *[projectName]***.dll** (**D**ynamic **L**ink **L**ibrary) not *[projectName]****.*exe** because the compiled code is loaded into other projects.

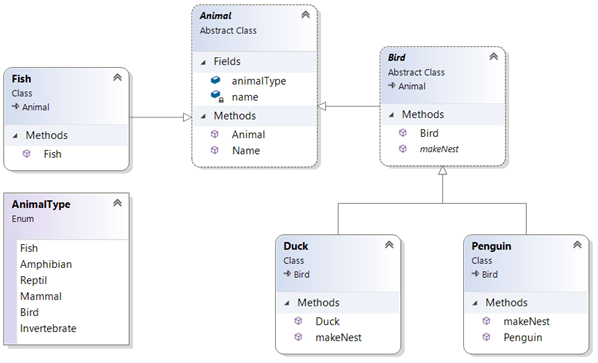
1. Open the Labs solution
2. Right-mouse click on the Solution and then select the   
   Add > New Project… menus
3. Type “class” in the search box and then select Class library project type like:  
   Graphical user interface, text, application

   Description automatically generated
4. Type **QALibrary** as project name and complete the rest of the steps
5. Delete the **class1** file as we don’t need this class
6. The **Labs** Console assembly (project) will use the QALibrary class library and therefore, it needs to reference it.   
   Please add a project reference to QALibrary like:





1. Please create the following classes inside the **QALibrary** project
2. Penguins and Ducks will be derived from the Bird class.   
   We have decided that "Bird" is too abstract for what we want.
3. The class diagram after completing the lab can be seen below.   
   You will find the full description of each class after the UML class diagram.



### 

### **Step by step**

1. Add the following classes to the **QALibrary** project
2. **Animal**
3. **Bird**
4. **Fish**
5. **Duck**
6. Create an **enum** to define six animal types like:  
   Please put this enum in its own file

**public enum** AnimalType {

***Fish***,

***Amphibian***,

***Reptile***,

***Mammal***,

***Bird***,

***Invertebrate***

}

1. Add code to the **Animal class**
2. Make this class **abstract**
3. Add a field to define the type, like:  
   **AnimalType** **animalType**;
4. Add a property procedure called **Name.** Make the set part private because once the name is set it should not change by any other class.
5. Provide this class with a constructor to set the animal’s Name
6. Feel free to add other properties and methods if you have time.
7. Add code to the **Bird class**
8. Make this class also **abstract**we want to make different kinds of birds, not a generic bird!
9. Bird should **Extends** (inherit from)the Animal class
10. Create a constructor to set a bird’s name
11. Set the type of the animal to Bird within the constructor.
12. Provide it with an abstract method called **MakeNest()** because all birds make their nest in a different way

Tip: **public** **abstract** **void** MakeNest();

1. Feel free to add other properties and methods if you have time.
2. Write code for the **Duck class**
3. **Extends** Bird  
   By default a **Duck** will also be an **Animal** Bird extends Animal)
4. You'll need to provide a constructor because the super class (Bird) has constructor with a String parameter to set the name.
5. Implement the **MakeNest()** method (just print a message)
6. Write code for the **Penguin class**
7. **Extends** Bird
8. Provide a constructor like the Duck class you created earlier
9. Create a class called **Fish**
   1. Set its name and type using a constructor  
      Feel free to give the fish other methods like **Swim()**
10. Open the class Program in the Labs Console project and make sure it is the start-up project (right-mouse click and then select *Set as Startup Project*)
11. Create a new static method called Lab9() below the Main() method
12. Call Lab9() from Main()
13. In the **Lab9()** method do:
14. Create a List of **Animal** type called **animals** (List<Animal>)
15. Create (instantiate) different Animal types and place them in the List of Animals
16. Write a ‘foreach’ loop that iterates over the **animals**.  
    This is an example of **Polymorphism**.

Every Fish, Penguin, Duck can be referred to as an Animal and can therefore be placed in a collection of type **Animal**.

1. Run your code to make sure it works
2. How would you detect if an animal in the list is a Bird and then call its **MakeNest()** method?
3. Test your code to make sure it works.
4. Add other animal types if you have time.

**\*\* End \*\***