Лабораторная работа №8

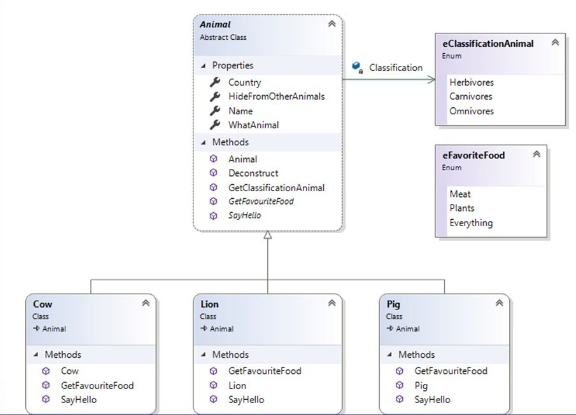
«Файловая система. Сериализация»

**Цели работы:**

1. Научиться работать с механизмом сериализации средствами языка C#.
2. Научиться работать с файловой системой средствами языка C#.

**Задание№1**

Используя диаграмму классов из лабораторной работы №7, реализуйте поддержку сериализации и десериализации класса Animal.



Создайте экземпляр класса Animal, инициализируйте все поля и выполните его Xml-сериализацию.

Добавьте в решение новый консольный проект, в котором десериализуйте класс Animal и выведите полученный объект на консоль.

**Задание№2**

Напишите приложение для поиска заданного файла во всех поддиректориях указанного пользователем пути.

Используйте FileStream для вывода содержимого файла на консоль.

Добавьте возможность сжатия найденного файла стандартными средствами .Net.

**Теоретические сведения**

**Сериализация/Десериализация**

Сериализация представляет собой процесс преобразования состояния объекта в форму, пригодную для сохранения или передачи. Дополнением к сериализации служит десериализация, при которой осуществляется преобразование потока в объект. Вместе эти процессы обеспечивают хранение и передачу данных.

В .NET доступны следующие технологии сериализации:

* При [двоичной сериализации](https://learn.microsoft.com/ru-ru/dotnet/standard/serialization/binary-serialization) сохраняется правильность типов, что полезно для сохранения состояния объекта между разными вызовами приложения. Например, можно обеспечить совместный доступ к объекту для разных приложений, сериализовав его в буфер обмена. Объект можно сериализовать в поток, на диск, в память, передать по сети и т. д. При удаленном управлении сериализация используется для передачи объектов "по значению" с одного компьютера или домена приложения на другой.
* При [сериализации XML и SOAP](https://learn.microsoft.com/ru-ru/dotnet/standard/serialization/xml-and-soap-serialization) сериализуются только открытые свойства и поля, а правильность типов не сохраняется. Этот метод полезен для предоставления или использования данных без ограничений работающего с ними приложения. Будучи открытым стандартом, XML привлекателен для совместного использования данных в Интернете. Аналогичным образом и SOAP представляет собой открытый стандарт, использование которого эффективно и удобно.
* При [сериализации JSON](https://learn.microsoft.com/ru-ru/dotnet/standard/serialization/system-text-json-overview) сериализуются только открытые свойства, а правильность типов не сохраняется. Будучи открытым стандартом, JSON привлекателен для совместного использования данных в Интернете.

Пример сериализации:



Пример десериализации:

