Лабораторная работа №7

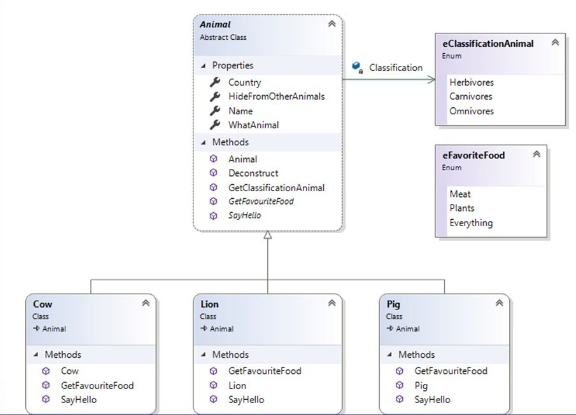
«Рефлексия»

**Цели работы:**

1. Научиться работать с механизмом рефлексии языка C#.

**Задание№1**

Создайте проект библиотеки классов со следующей диаграммой классов



В созданном проекте реализуйте класс пользовательского атрибута со свойством Comment. Примените атрибут с произвольным комментарием к каждому классу диаграммы классов.

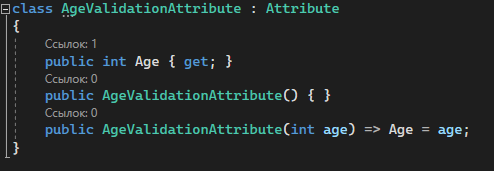
Создайте приложение, которое будет подключать созданную библиотеку классов и средствами рефлексии генерировать файл xml-представления всей диаграммы классов библиотеки, в том числе с созданными пользовательскими атрибутами.

**Теоретические сведения**

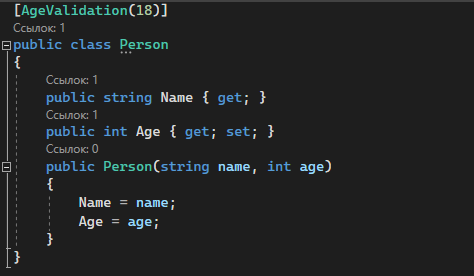
**Атрибуты**

Атрибуты в .NET представляют специальные инструменты, которые позволяют встраивать в сборку дополнительные метаданные. Атрибуты могут применяться как ко всему типу (классу, интерфейсу и т.д.), так и к отдельным его частям (методу, свойству и т.д.). Основу атрибутов составляет класс System.Attribute, от которого образованы все остальные классы атрибутов. В .NET имеется множество встроенных классов атрибутов. И также мы можем создавать свои собственные классы атрибутов, которые будут определять метаданные других типов.

Допустим, нам надо проверять пользователя на соответствие некоторым возрастным ограничениям. Создадим свой атрибут, который будет хранить пороговое значение возраста, с которого разрешены некоторые действия:



Теперь применим его к некоторому классу:



класс Person применяет атрибут. Для этого имя атрибута указывается в квадратных скобках непосредственно перед определением класса. Причем суффикс Attribute указывать необязательно. Обе записи [AgeValidation(18)] и [AgeValidationAttribute(18)] будут равноправны.

Если конструктор атрибута предусматривает использование параметров (public AgeValidationAttribute(int age)), то после имени атрибута мы можем указать значения для параметров конструктора. В данном случае передается значение для параметра age. То есть фактически мы говорим, что в AgeValidationAttribute свойство Age будет иметь значение 18.

В качестве альтернативы можно использовать именованные параметры для всех свойств атрибута, если класс атрибута имеет конструктор без параметров: [AgeValidation(Age = 18)]

Теперь получим атрибут класса Person и используем его для проверки объектов данного класса:

