Задание на серверную разработку

Задание Локализация

Для desktop-приложения нужно разработать набор классов для работы с локализованными строками, которые будут отображаться в интерфейсе пользователя.

Требования:

- Локализованное значение строки должно быть доступно по коду строки локализации и переданному объекту CultureInfo.
- Если при запросе локализованного значения строки объект CultureInfo не был передан, то необходимо использовать текущую культуру потока.
- В качестве источника строк могут выступать как ресурсы текущей сборки, так и сторонние источники (БД, XML-файл или иное). Необходимо предусмотреть возможность подключения произвольного источника строк.
- Система локализации должна самостоятельно определять источник строк по переданному коду строки локализации (коды могут быть любыми).
- Разные источники могут иметь пересечения по кодам строк, система должна корректно определять из какого источника брать строку в каждом конкретном случае.
- Предусмотреть возможность хранения строк локализаций в полях класса без привязки к конкретной культуре с возможностью последующего получения значения строки для нужного языка.

Реализация

Для работы с локализациями необходимо разработать класс LocalizationFactory, который предоставляет следующие методы:

- GetString – возвращает значение строки локализации по её коду для переданной культуры;
- RegisterSource – регистрирует источник строк локализаций.

Задание на анализ кода

Дан исходный код функции. Функция выполняет поиск значения переданного атрибута элемента Xml-документа.

На вход функции передается путь до Xml-файла; название элемента (в котором ищется значение); название атрибута, значение которого необходимо получить.

Необходимо указать на недостатки и возможные ошибки исходного кода и предложить свою реализацию алгоритма.

```
static string Func1(string input, string elementName, string attrName)
{
  string[] lines = System.IO.File.ReadAllLines(input);
  string result = null;

  foreach (var line in lines)
  {
    var startElEndex = line.IndexOf(elementName);

    if (startElEndex != -1)
    {
      if (line[startElEndex - 1] == '<')
      {
```

```
        var endElIndex = line.IndexOf('>', startElEndex - 1);
        var attrStartIndex = line.IndexOf(attrName, startElEndex, endElIndex - startElEndex + 1);

        if (attrStartIndex != -1)
        {
          int valueStartIndex = attrStartIndex + attrName.Length + 2;

          while (line[valueStartIndex] != '"')
          {
            result += line[valueStartIndex];
            valueStartIndex++;
          }

          break;
        }
      }
    }
  }

  return result;
}
```