Техническое задание на разработку парсера

Требуется создать парсер доски объявлений по разделу «Недвижимость», который должен выбрать все последние объявления, начиная с определенного объявления. Парсер должен быть написан на C# и выполнен в видеDLL библиотеки. Парсер получает в качестве входящих данных рубрику, регион, действие(продают, покупают и т. д.) и идентификатор объявления. Возвращаемый результат должен содержать все последние объявления размещенные на сайте после объявления, идентификатор которого передан входящим параметром. Если идентификатор последнего входящего объявления пустой, то возвращаемый результат - самое последнее объявление и его идентификатор.

Программа парсер – это **ОДНА** ваша библиотека, **ОДНА** ваша бд sqlite + **СОГЛАСОВАННЫЕ** сторонние библиотеки и бд. Согласованные библиотеки и БД должны быть перечислены в файле с именем библиотеки и расширением LST (имя файла — комментарий и назначение).

Используемые бд:

- БД справочников регионов, рубрик, действий в формате Sqlite
- Название БД должно совпадать с названием библиотеки
- Справочники редактированию и изменению в одностороннем порядке без согласования не подлежат.
- Все возвращаемые библиотекой рубрики, регионы и действия должны соответствовать структуре справочников. Если структура каталогов ресурса отличается, то она должна быть приведена к структуре справочников.

Используемые технологии, библиотеки:

- VS 2013 + ReSharper
- .NET 4.5 (использование async/await)
- Managed Extensibility Framework (http://msdn.microsoft.com/ru-ru/library/dd460648(v=vs.110).aspx)
- HtmlAgilityPack – пакет из Nuget для парсинга Html
- servicestack.ormlite.sqlite32– пакет из Nuget для работы с Sqlite БД
- RT.Crawler - библиотека для получения контента с веба (вам предоставится исходный код)
- RT.ParsingLibs – базовые классы для парсинга, абстрактный модуль парсера (вам предоставится библиотека)

Требования к библиотеке парсера:

- Должна строиться по аналогии с примерами бибилотек RT.ParsingLibs.AbstractFirst и RT.ParsingLibs.AbstractSecond
- Парсер должен реализовать интерфейс IParsingModule из библиотеки RT.ParsingLibs
- Библиотека содержит следующие методы (IParsingModule):

1. **About** — возвращает информацию и координаты разработчика, а так же информацию о передаче исключительных прав. Текст о передаче исключительных прав будет выслан дополнительно.

2. **Sources** – метод получает «рубрика, категория, регион», проверяет обрабатывается ли такой набор исходных данных данной библиотекой и возвращает коллекцию названий ресурсов (сайтов) в случае возможной обработки.
3. **`KeysRubrics`** – возвращает коллекцию «ИД рубрик» в структуре сервиса, которые умеет обрабатывать библиотека
4. **`KeysRegions`** – возвращает коллекцию «ИД регионов» в структуре сервиса, которые умеет обрабатывать библиотека
5. **`KeysActions`** – возвращает коллекцию «ИД действий» в структуре сервиса, которые умеет обрабатывать библиотека
6. **Result** – получает рубрику, категорию, регион, ID и DataTime и Хеш-MD5 последнего объявления. Результат содержит все последние объявления размещенные на сайте после ранее полученного последнего объявления, ID и DataTime и Хеш-MD5 которых является входящими параметрами метода . Если параметры последнего входящего объявления пусты, то возвращаемый результат - самое последнее объявление.

- Все выходные данные с библиотеки должны быть по максимуму заполнены:

    Public property Address Адрес

    Public property AppointmentOfRoom Назначение помещения

    Public property CostAll Цена за весь объект

    Public property CostPerMeter Цена за метр

    Public property District Район

    Public property Floor Этаж

    Public property FloorNumber Количество этажей

    Public property Furnish Отделка

    Public property IsLoggia Балкон/лоджия

    Public property IsParking Паркинг во дворе или доме

    Public property KitchenSpace Площадь кухни

    Public property LandSpace Площадь участка, соток

    Public property LeasableSpace Сдаваемая площадь

    Public property LivingSpace Жилая площадь

    Public property RealEstateType Тип недвижимости

    Public property RoomNumber Количество комнат

    Public property Tenancy Срок аренды

    Public property TotalSpace Общая площадь

    Public property ViewFromProperty Вид из окон

    Public property WallMaterial Материал стен

    Public property Wc Санузел

- Получения контента с веба вынести в библиотеку RT.Crawler (добавление, без изменения существующего кода)
- Должен быть создан отдельный проект с unit-тестами MSTest
- Одна библиотека должна обрабатывать 1 ресурс
- Вся кодировка устанавливается в UTF8.
- Все исходники библиотеки должны быть открыты.
- Разработчик библиотеки должен обладать на них авторскими правами и передать исключительные права на использование кодов.
- Если в качестве региона задана область, то объявления возвращаются по загородной недвижимости.
- Одним из условий работы парсера является требование к оптимизации алгоритма получения новых объявлений, т.е. необходимо делать минимальное количество запросов к ресурсу. Для этого парсер должен использовать форму расширенного поиска, чтобы получать выборку нужных объявлений. Затем отсеивать объявления, которые были до IdRes и использовать дальнейшие ссылки только для новых объявлений. Если в выборке полученной от расширенной формы нет объявления с IdRes, то возвращать не более CountAD объявлений и вернуть соответствующий код парсинга
- Если в справочнике DLL указана одна подрубрика, а на сайте она разделена на две, например, в DLL “Участок, сад, дача», а на свйте это две отдельные рубрики: «Земля» и «Дача», то алгоритм должен быть следующим (думаю понятно, почему «Участок»= «Земля»): Сначала делаем запрос по первой рубрике «Земля». По ТЗ получаем объявления и idRes по земле. Затем делаем запрос по второй рубрике «Дача». По ТЗ получаем объявление и idRes по дачам. В возвращаемом результате выдаем склееный idRes через спецсимвол (idRes = idRes по земле + “&”+ idRes по дачам. Спецсимвол и последовательность очередности обхода рубрик разработчик выбирает сам. В качестве объявления выдается самое последнее объявление по времени обновления (публикации если время обновления установить невозможно) или по id, если время установить невозможно. При получении входящим параметром idRes, от по спецсимволу разделяется на idRes по земле и idRes по дачам. Каждая ветвь выполняется со своим idRes. Выходящий параметр содержит склейку обновленных idRes и коллекцию объявлений каждой ветви соответствующим условию ТЗ. Если на сайте рубрик больше, то тот же самый алгоритм и склейка idRes по колличеству ветвей.
- Если в справочнике DLL есть 2 разных рубрики, например, первая «Дача и коттедж» и вторая «Таунхаусы», а на сайте это содержится в одной рубрике «Дача, коттедж, Таунхаус» (или вложены в эту рубрику, например, «Таунхаусы» является одной из подрубрик « Дача и коттедж»), то при выдаче результата исключающие рубрики должны быть исключены. Например, при выборе рубрики в DLL «Дача и коттедж» в результатах должны быть исключены объявления по «Таунхаусы».
- при тестах обратите внимание на возможную ошибку : Парсер может выдавать не последнее (или не все последние) объявление. На сайте по этому запросу могут быть более позднее объявление (по дате). Так может получиться, если парсер выбирает последнее объявление по №. Если автор обновил дату публикации старого объявления, т.е. его № меньше, чем скажем предпоследнего, то парсер его не видит. В этом случае нужно сортировку поменять не по номеру, а по дате и времени обновления(публикации)

  Пример:

Представим ситуацию, когда

было id=1 data=10:00;

разместили id=2 data=11:00 тут LastPublicationId=id1=2 data1=11:00;

обновили id=1 data=12:00;

разместили id=3 data=13:00;

обновили id=2 data=14:00;

Если выбирать последнее объявление по №, то выпадает объявление с id=1 и id=3

если у объявлений есть явные дата и время, то лучше выбрать в качестве LastPublicationId = Data(с учетом времени)

если входящий LastPublicationId = “”, то LastPublicationId = Data

если входящий LastPublicationId = Data, то выбираем все у которых дата > Data, но количество выбранных не более CountAD.

Требования к вашей бд:

- Один файл в формате sqlite
- БД не должна по результатам работы программы увеличиваться в размерах.
- Все таблицы соответствия должны быть заполнены и приведены к единой структуре.
- Заполнение таблиц соответствия осуществляет разработчик.

Требования к сторонним библиотекам:

- По возможности загружать из хранилища Nuget

Пример реализации парсеров можно найти в библиотеках:

- RT.ParsingLibs.AbstractFirst (вам предоставится исходный код)
- RT.ParsingLibs.AbstractSecond (вам предоставится исходный код)

Требования к документации:

- Комментарии в коде
- Сложные алгоритмы должны быть описаны в отдельном документе docx

Термины

ID объявления - уникально в пределах ресурса и составляется самим ресурсом. Назовем его IdRes. Этот IdRes нам нужен для того, чтобы контролировать с какого ранее найденного объявления нам нужно получить более свежие объявления. Данный IdRes может быть числом, датой и временем размещения объявления или ссылкой. В пределах ресурса IdRes может быть скрыт от нас и нам не известен, поэтому мы должны сами идентифицировать эти объявления. Что выбрать в качестве IdRes на ресурсе определяет сам разработчик, который реализует парсинг ресурса. При первом вызове метода парсера из DLL не известно о IdRes объявления внутри ресурса, поэтому он передает значение NULL. В этом случае парсер должен выбрать самое последнее объявление и вернуть его IdRes ресурса. Если IdRes не найден, то парсер возвращает последнии CountAD объявлений (такая ситуация может получится, если размещенное объявление к следующей итерации будет удалено на ресурсе) Т.к. источников много и заранее не известен каким будет IdRes, то для IdRes зарезервировано 3 переменных ID и DataTime и Хеш-MD5 последнего объявления. Каким из них пользоваться решает разработчик.

Регион – это населенный пункт (мир, страна, федеральный округ, область, город, поселок …). Справочник регионов в конечном виде имеет древовидную структуру.

Рубрика (каталог) - содержит только те рубрики, которые сервис умеет обрабатывать

Действие (категория) – куплю, продам, обменяю, арендую и т.д.

Бинд – это тройка идентификаторов региона, рубрики и действия