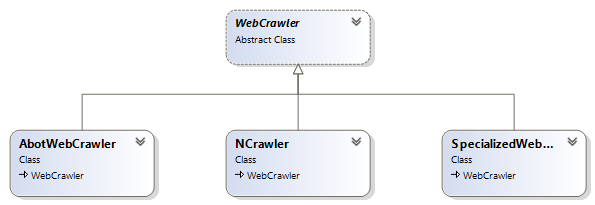
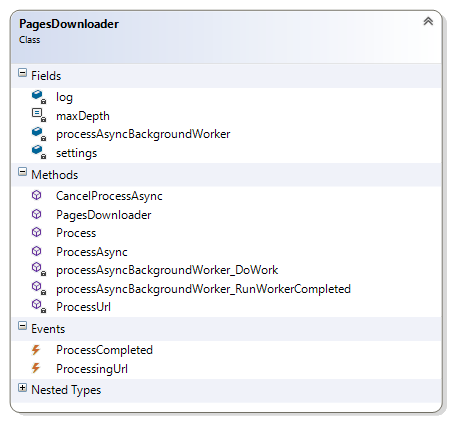
Архитектура

Архитектура разработанного проекта выглядит следующим образом:

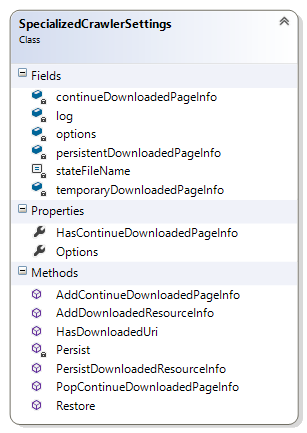


WebCrawler является базовым абстрактным классом. На данном этапе содержит только метод Crawl. Добавлены две библиотеки: AbotWebCrawler и NCrawler для дальнейшего сравнения с прототипом – SpecializedWebCrawler.

Основной процесс работы поискового робота реализован в классе PagesDownloader.

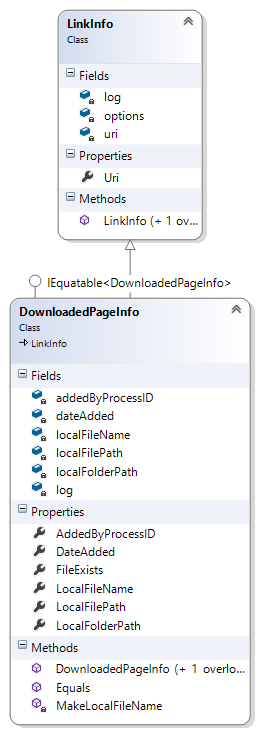


В конструкторе PagesDownloader(CrawlerOptions options) инициализируется логгер (log4net), ресторятся сеттинги SpecializedCrawlerSettings – данный класс отвечает за «состояние» кролера.

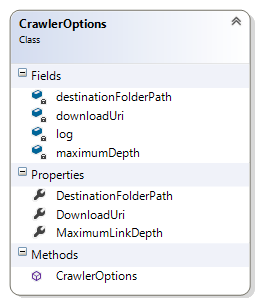


В классе SpecializedCrawlerSettings в специальный файл производится запись/считывается информация о состоянии кролера. А именно, информация о полностью обработанных страницах, информация о страницах, которые были обработаны не полностью (например, работа кролера была остановлена в процессе извлечения ссылок) – такие страницы будут обработаны заново. Также в данном файле содержится информация о страницах, с которых кролеру следует начать работу.

Под «информацией о странице» подразумеваются следующие данные:



Вернемся к описанию класса PagesDownloader. Как было сказано раньше параметром конструктора является CrawlerOptions.

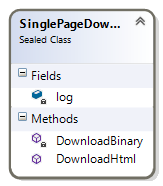


DestinationFolderPath – путь к папке, в которую будут сохраняться скачанные страницы и файл с состоянием кролера.

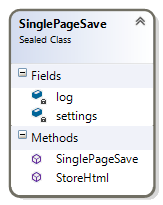
DownloadUri – uri страницы, с которой следует начать работу кролера, если файл состояние таковой не содержит.

MaximumLinkDepth – максимальная глубина работы кролера (по умолчанию 5).

Главные методы класса PagesDownloader – Process(), ProcessAsync() для последовательной и асинхронной обработки. Прежде чем описывать последовательность работы кролера, рассмотрим еще два реализованных класса:



SinglePageDownload – класс для преобразования страницы с заданным Uri и строку.



SinglePageSave – класс для создания локального файла и сохранения в него строки, загруженной с помощью класса SinglePageDownload.

Теперь можно вернуться к описанию метода Process() класса PagesDownloader:

1. Проверить содержит ли файл состояния кролера ссылки, с которых следует начать работу. Если не содержит, то начать со ссылки заданной в конструкторе.
2. Пока файл состояний содержит ссылки, с которых следует начинать работу, делать следующее:
   1. Убрать из списка ссылок обрабатываемую ссылку, сохранив текущее состояние кролера в файл.
   2. Проверить, что обрабатываемая ссылка имеет глубину меньше максимальной.
   3. Проверить, что ранее эта страница не обрабатывалась, почекав файл состояния кролера.
   4. Считать страницу в текстовую переменную textContent.
   5. Сохранить textContent в локальный файл.
   6. Добавить информацию о странице в список обрабатываемых и сохранить состояние кролера в файл.
   7. ----------------Машин стафф с парсингом-----------------------
   8. Повторить пункты 2.2 – 2.8 для каждой найденной ссылки.

Кролер вызывается из метода Main() следующим образом, с указанием ссылки, с которой следует начать работу:

WebCrawler crawler = new SpecializedWebCrawler();

crawler.Crawl(@"http://spbu.ru/");