Софийски университет „Св. Кл. Охридски”

Факултет по математика и информатика

Бакалавърска програма
„Софтуерно инженерство”

Предмет: XML технологии за семантичен Уеб

Зимен семестър, 2021/2022 год.

Тема №63: „Генериране на XML по DTD документ със съдържание от Wikipedia“

Курсов проект

Автори:

Мирослав Дионисиев, фак. номер 62390

Павел Сарлов, фак. номер 62393

януари, 2022 г.

София

Съдържание

1 Въведение

Текущият документ описва нашия проект, който представлява XML генератор на съдържание от *Wikipedia* статия по подадена от потребителя DTD граматика. Интуитивно човек ще си зададе въпроса „Как от нещо, което се използва за валидация на друго, може да се създаде това другото нещо?“. Това не е тривиална работа и не съществуват подобни инструменти. Целта на проекта е да реши точно този проблем като предостави извличане на конкретна информация по дадени критерии в XML файл.

За реализиране на проекта е използван обектно-ориентирания език *C#* и средата за разработка *Visual Studio*. Причина за избора е изобилието от удобни инструменти и библиотеки за реализация, например библиотеката *System.Xml*, която предоставя удобни средства за изграждане на XML DOM дърво и валидирането на генерирания XML документ.

В теущия документ са описани следните:

- работен процес на приложението и съдържание на неговия резултат
- дизайн на системата
- начин на тестване на системата
- поглед в бъдещето развитие на приложението
- разпределение на работата между участниците в екипа
- използвани източници

2 Анализ на решението

2.1 Работен процес

Входно съдържание/данни, които потребителя подава на системата:

- Валидна DTD граматика, по която се генерира XML документ
- URL адрес към *Wikipedia* статия

Обработка на данните:

- Подадената граматика се валидира
- Създават се обекти, които симулират елементите и атрибутите в граматиката
- От *Wikipedia API* се взима съдържанието на статията и се създава XML дърво от него
- Създаденото XML дърво се обхожда и се проверява за несъответствия с граматиката:
    - Ако има елементи, които не присъстват в граматиката, но са налични в XML дървото, се премахват заедно с техните деца

        - Ако има елементи, които присъстват в граматиката, но не са налични в XML дървото, се създава скелет на подадената от потребителя граматика

    - От финалното XML дърво се генерира XML файл

2. Изходно съдържание/данни, които потребителя получава от системата:

    - Крайният XML файл се подава на потребителя за корекции и/или изтегляне

2.2 Структура на съдържанието

Структурата на изходния XML и на DTD граматиката съответства на структурата на *Wikipedia* статията. Имената на елементите съвпадат с заглавията в статията и следват тяхната йерархия.

2.3 Тип и представяне на съдържанието

Проектът е предоставя среда за генериране и обработка на XML съдържание както и неговото изтегляне във файл. Съдържание е в UTF-8 кодиране и използва XML версия 1.0. За униформеност и лесната имената на елементите и атрибутите са изписани с главни букви, тоест няма значение как са изписани в DTD граматиката, важно е единствно да съвпадат с имената на съдържанието от статията.

3 Дизайн

1. Модул за четене/писане на файлове

    - *FileManager* – предоставя интерфейс за четене и писане във файл както синхронно, така и асинхронно.

2. Модул за взимане на информация от *Wikipedia*

    - *WikiScrapper* – предоставя интерфейс *GetContent* за получаване на съдържанието на *Wikipedia* статия по даден URL адрес. Съдържанието се запазва в речник, чийто ключ е заглавието на параграфа, а стойността е кортеж от дълбочината в йерархията и съдържанието на параграфа. Интерфейсът *GetXml* позволява съдържанието да се форматира като XML DOM дърво.

3. Модул за генериране на валиден XML по DTD

    - *XMLGenerator* – приема подадената от потребителя DTD граматика. Интерфейсът *GetXMLFromWikiTextAsync* приема URL адреса на *Wikipedia* статията и създава базово XML DOM дърво. Това дърво се валидира и редактира спрямо подадената граматика. В случай на несъответствия в настоящото дърво се генерира XML скелет по DTD граматиката.

4. Модул за четене на DTD граматика

- *DTDFileReader* – приема подадената от потребителя DTD граматика и създава списъци от обекти описващи елементите и атрибутите в граматиката със съответните им свойства.

5. Модул за валидация на XML по DTD

    - *XMLValidator* – използва методите на библиотеката *System.Xml*, за да валидира крайния XML документ.

6. Модул за потребителския интерфейс

    - Използва се библиотеката *CodeMirror* за предоставяне на текстови блокове за въвеждане и редакция на текстово съдържание. Лявият текстов блок е за входната DTD граматика, а десният – за изходния XML файл. При въвеждане във всеки от блоковете се изпълнява събитие за валидиране на промените като се извиква съответния валидатор от backend-а.

    - Между двата текстови блока има поле за въвеждане на URL адреса на *Wikipedia* статията, както и бутони за генериране и изтегляне.

7. Други

    - *DTDValidator* – използва методите на библиотеката *System.Xml*, за да валидира формалността на DTD граматиката.

    - *Element* – клас описващ името и съдържанието на елемента, неговите деца и честота на техните срещания.

    - *Attribute* – клас описващ името на атрибута, вида на стойността, списъкът от възможни стойности и т.н.

    - *Logger* – предоставя интерфейс за записване на системни съобщения.

    - *TypesConverter* – предоставя конвертиране между низове, байтове, потоци и DOM дървета.

4 Тестване

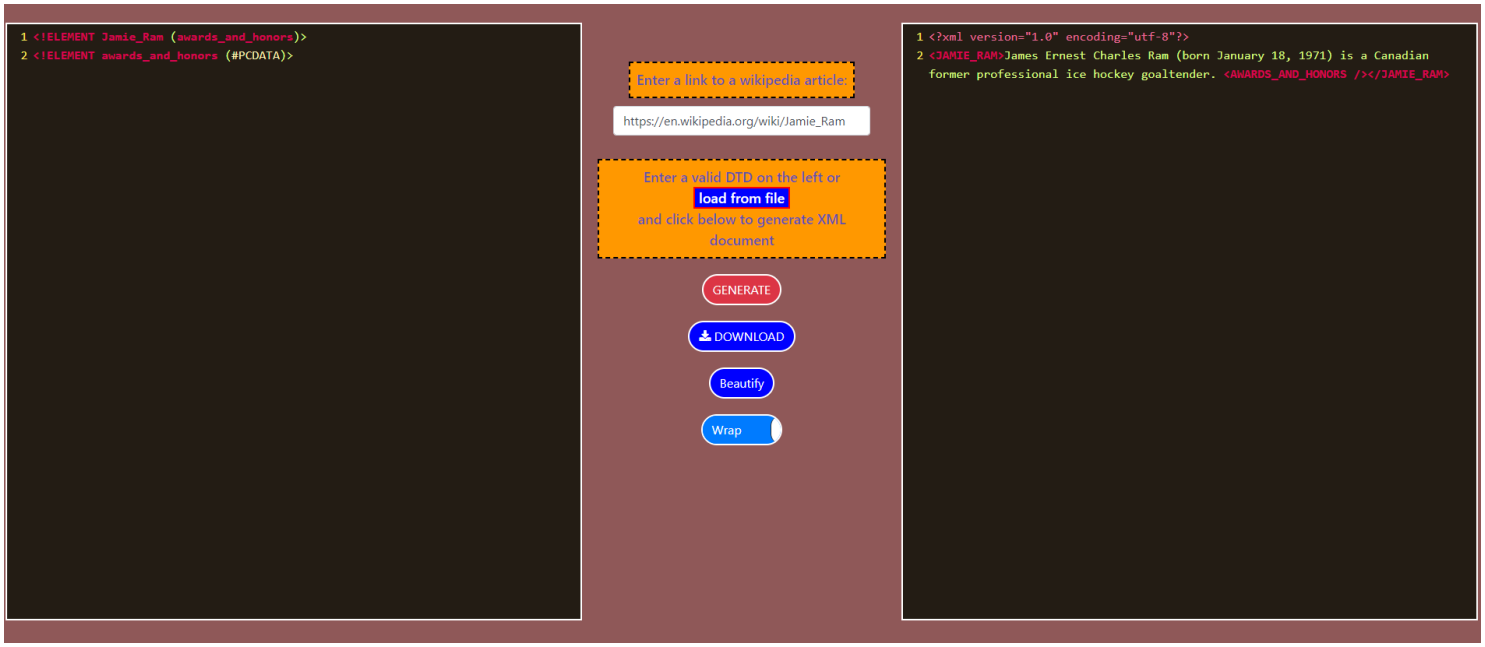
Извършени са тестове с валидни и невалидни DTD диаграми, с успешна генерация на XML съдържание и генериране на XML скелет и на невалиден URL адрес на *Wikipedia* статия. Тествани са само модерни браузъри като *Firefox, Edge, Chrome*.

Следните снимки показват част от потребителските случаи:

4.1 Тест с DTD граматика, отговаряща на *Wikipedia* съдържанието

4.2 Тест с DTD граматика, неотговаряща на *Wikipedia* съдържанието

4.3 Тест с DTD граматика, отговаряща на *Wikipedia* съдържанието, която взима само определени елементи



5 Заключение и възможно бъдещо развитие

Задачата се оказа доста предизвикателна, тъй като при парсването на DTD граматиката има изключително много случаи за разглеждане. Разбира се, не сме успяли да покрием всеки един от тях, но достатъчно.

Като наблюдения след завършване на проекта може да се отбележат:

- Генерирането на DOM дърво по предварително предоставена граматика не е тривиален процес, защото граматиката е предназначена за валидиране на XML съдържание, а DOM дървото описва съществуваща XML йерархия и предоставя възможност за лесното ѝ редактиране. Обратния процес се сблъсква с липса на инструменти за осъществяването му.
- Въпреки пречките, *C#* се оказа изключително удобен език за целта, тъй като предоставя много инструменти и библиотеки за работа с DOM дървета, както и реализиране на логиката на приложението.

Проектът може да се развие и да генерира XML файлове от различни URL адреси освен *Wikipedia*. Може да се подобри вътрешната логика и да се добави обработка на единици и други DTD декларации. Също така, да се разгледат различни случаи на въведената граматика, за да се увеличи покритието на парсера. Допълнително, може да се напишат *unit tests*, за което не е достигнало време при разработката на проекта.

6 Разпределение на работата

- Мирослав Дионисиев
    - Модул за генериране на валиден XML по DTD
    - Модул за четене на DTD граматика
    - DTD модели на елементите и атрибутите
    - Цялостната логика по backend-а на системата
- Павел Сарлов
    - Модул за четене/писане на файлове
    - Модул за взимане на информация от *Wikipedia*
    - Модул за валидация на XML по DTD
    - Модул за потребителския интерфейс
    - *DTDValidator* и *Logger*

Съвместна работа по цялата структура на проекта.

7 Използвани литературни източници и Уеб сайтове

- Документация на System.Xml
- Документация на CodeMirror