

ДОКУМЕНТАЦИЯ НА ПРОЕКТА

Функционалност, специфична за JSON формата, която няма директен аналог в XML

[JSON Patch и XML Patch – Демонстрация и сравнение](#)

1. Описание на проекта

Проектът има за цел да демонстрира функционалност, специфична за JSON формата, която няма директен аналог в XML. Това е частичното обновяване на данни чрез JSON Patch – официален стандарт, дефиниран в RFC 6902.

JSON Patch позволява модифициране на JSON документи чрез последователност от операции, без необходимост от преразписване на целия документ. Така се постигат ефективни актуализации, които намират широко приложение в REST API, конфигурационни системи, синхронизация на данни и др.

XML от своя страна няма стандартен, широко приет формат за описване на частични промени върху XML документ. Единственият начин за демонстрация е чрез разработване на custom XML Patch формат, който имитира JSON Patch операциите. Това ясно показва концептуалните разлики между двета формата и защо JSON предоставя по-добра поддръжка за частични актуализации.

2. Техническо обяснение

Проектът представлява конзолно приложение на C# (.NET), което зарежда входни JSON и XML документи, прилага patch операции върху тях и генерира изходни „patched“ версии.

Техническият подход включва:

2.1. За JSON:

- Използва се System.Text.Json и JsonNode за динамична обработка на JSON гърбомо.
- Patch документът се десериализира като списък от операции.
- Операциите се прилагат последователно чрез методи add, remove и replace.
- Адресирането се извършва чрез JSON Pointer.

2.2. За XML:

- Използва се XDocument и XPathSelectElement за намиране на елементи.

- Patch документът е custom XML формат, дефиниран специално за проекта.
- Операциите add, remove и replace се изпълняват чрез модификация на XML сървото.
- Адресирането става чрез XPath.

Тази архитектура показва:

- JSON Patch е стандартизиран и универсален.
- XML изисква custom решения, тъй като стандартна алтернатива липсва.

3. Описание на JSON Patch имплементацията

JSON Patch имплементацията се състои от три основни части:

- 3.1. Клас JsonPatchOperation – представлява една операция от JSON Patch масива. Включва:
 - Op – вид на операцията (add, remove, replace)
 - Path – JSON Pointer път
 - Value – стойност (за add и replace)
- 3.2. Клас JsonPatchApplier – съдържа логиката за прилагане на Patch-а.
 - Apply – приема целевия JSON и списък от операции
 - ApplyOperation – интерпретира операцията и определя какво трябва да се промени
 - Add, Remove, Replace – методи за модификация на JsonObject и JSONArray
- 3.3. Метод RunJsonPatch – зарежда файловете user_full.json и patch_full.json, прилага операциите и записва резултат в user_full_patched.json.

Основна характеристика на JSON Patch е, че може да прилага операции върху:

- Вложени обекти
- масиви
- динамични структури без фиксирана схема
- произволни пътища чрез JSON Pointer

Това го прави идеално подходящ за частично обновяване в реални информационни системи.

4. Описание на XML Patch имплементацията

XML Patch частта е реализирана като custom решение, тъй като няма стандарт.

Състои се от:

- 4.1. Custom patch файл patch_full.xml, съдържащ операции:
 - <Replace path="..." value="..."/>
 - <Remove path="..."/>
 - <Add path="...">>...</Add>
- 4.2. Клас XmlPatcher, който обработва документа:
 - Apply – зарежда XML файла и patch файла, обхожда операциите и прилага:
 - Replace – променя стойност на елемент
 - Remove – премахва елемент от дървото
 - Add – добавя нов елемент в посочен възел
 - ApplyReplace – заменя текстовата стойност
 - ApplyRemove – премахва възела
 - ApplyAdd – добавя дъщерни елементи към целевия възел

Агресирането в XML се извършва чрез XPath изрази, например:

/User/Settings/Privacy/Ads

Това показва, че XML модификациите изискват по-сложни операции, по-малко стандартизиирани и по-трудни за автоматизиране в сравнение с JSON.

5. Описание на основните класове и методи

- 5.1. Основни класове:

- JsonPatchOperation – описва JSON Patch операция.
- JsonPatchApplier – изпълнява JSON Patch логиката.
- XmlPatcher – изпълнява custom XML Patch логиката.
- Program – управлява менюто и извиква JSON/XML обработките.

- 5.2. Основни методи:

- Apply(JsonNode, IEnumerable<JsonPatchOperation>) – прилага JSON Patch.
- ApplyOperation(JsonNode, JsonPatchOperation) – обработва една операция.
- Add/Remove/Replace – модифицирам JSON дървото.
- Apply(string, string, string) – прилага XML Patch.
- ApplyReplace/ApplyRemove/ApplyAdd – XML операции.

6. Примерен flow на изпълнение

- 6.1. Потребителят стартира програмата.
- 6.2. Появява се меню:
 1. Прилагане само на JSON Patch
 2. Прилагане само на XML Patch
 3. Прилагане на двете
- 6.3. Програмата зарежда входни файлове:
 - user_full.json / patch_full.json
 - user_full.xml / patch_full.xml
- 6.4. JSON Patch се прилага чрез JsonPatchApplier.
- 6.5. XML Patch се прилага чрез XmlPatcher.
- 6.6. Получават се файлове:
 - user_full_patched.json
 - user_full_patched.xml
- 6.7. Показва се съобщение за успешна операция.

7. Разлики между JSON Patch и XML Patch

- 7.1. JSON Patch е стандартизиран (RFC 6902). XML Patch стандарт няма.
- 7.2. JSON използва JSON Pointer за адресиране. XML използва XPath.
- 7.3. JSON Patch работи естествено с масиви. XML няма концепция за масив.
- 7.4. JSON Patch е лек, кратък и лесен за четене. XML Patch е по-обемист и по-сложен.
- 7.5. XML обработката изисква custom имплементация.
- 7.6. JSON Patch е широко използван в уеб API-тата, XML Patch – почти никъде.

8. Изводи от курсовата работа

В резултат на реализираната демонстрация ясно се вижда, че:

1. JSON Patch предоставя мощен и удобен механизъм за частични обновявания.
2. XML няма стандартен механизъм за patch операции.
3. XML Patch трябва да бъде изцяло custom, което води до по-сложен и труден за поддръжка код.
4. JSON е по-подходящ формат за динамични структури и чести актуализации.

5. Реализацията доказва на практика, че JSON предлага функционалност, която XML не може да възпроизведе по стандартен начин.

Това напълно покрива изискването на заданието и показва разликите между двата формата чрез реален код и реалистични примери.