Vol. 4, No. 1, 2022

DOI: https://doi.org/10.23939/csn2022.01.091

УДК 004.632

ПРИНЦИПИ СТВОРЕННЯ ВІЗУАЛЬНОГО РЕДАКТОРА XML- ТА JSON-ФОРМАТІВ

І. П. Кутник, О. Л. Лашко

Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин
E-mail авторів: ivan.kutnyk.ki.2018@lpnu.ua, oksana.l.lashko@lpnu.ua

У цій статті досліджено особливості сприйняття великих обсягів текстової інформації та проаналізовано потреби у візуальному редагуванні. Наведено реалізацію програмного продукту, що працює із XML- та JSON-форматами і забезпечує графічне та кольорове виділення основних елементів синтаксису.

Розроблено структуру програми, та розділено її на декілька модулів: зчитування файлів, аналіз вмісту та відображення. Спроєктовано загальний алгоритм роботи програми. Створено функціонал для відкриття файлу та подальшого його аналізу в програмі. Представлено візуалізацію прочитаного та проаналізованого файлу.

Метою статті є відображення результатів дослідження проблеми візуалізації даних при обробці інформації користувачами, а також висвітлення результатів реалізації програмного продукту, який забезпечує можливість редагувати, створювати, зберігати вміст файлів XML- та JSON-форматів, та підтримує кольорове і графічне розділення елементів програмного тексту. При цьому гарантується до 7 одночасно відкритих файлів з кількістю рядків у файлі – до 1000.

Ключові слова: зчитування файлу, аналіз вмісту, візуальне редагування, JSON, XML.

Вступ

Інформація виступає основним об'єктом сучасного суспільства, і її роль сьогодні важко переоцінити. На теперішньому етапі життя відбувається черговий вибух технологічної та мирної соціальної революції – становлення інформаційного суспільства, що є передумовою для еволюційного переходу до наступної стадії розвитку людства, технологічною основою якої є індустрія створення, обробки і передачі інформації [1].

Розвиток теперішнього світу вимагає значної візуалізації інформації, оскільки обсяги її значно збільшились. Виникають проблеми зі створенням, переглядом або редагуванням даних.

Візуальне редагування – це форма комп'ютерного редагування тексту, що часто використовується в програмуванні, яка відображає зміни негайно, через візуальний інтерфейс, а не покладаючись на зміни рядків і відкладене відображення цих змін. Це тип редагування, який зустрічається в сучасних комп'ютерних програмах, включаючи програми обробки текстів і створення документів, а також прості текстові програми, як-от редактори звичайного тексту. На відміну від рядкових редакторів, програми візуального редактора дозволяють користувачеві швидше бачити, що робиться, і які зміни він чи вона вносить у документ. І хоча візуальне редагування часто асоціюється з комп'ютерним програмуванням і створенням програм, воно також часто використовується при створенні текстових документів з багатьох інших причин.

Візуальне редагування поширене у програмуванні, оскільки воно дозволяє програмісту бачити зміни, коли вони відбуваються. Якщо буде допущена синтаксична помилка або натиснута

неправильна клавіша, програма візуального редактора негайно відобразить цю помилку, і її буде легше вловити перед компіляцією та запуском програми. Візуальне редагування також є дуже корисним для письменників та інших користувачів програм обробки текстів, оскільки дозволяє людині швидше виявляти та виправляти граматичні помилки або слова з помилкою [2].

1. Аналіз останніх досліджень та публікацій

Щодня кожного з нас оточують величезні об'єми різноманітного контенту. Тисячі статей, які ніхто не читає, звіти з сотнями сторінок, які переглядають «по діагоналі», текстові презентації, які не викликають зацікавленості. Мозок змушений ігнорувати значну частину інформації, щоб вберегти організм від перенапруження [1–2].

Перші відомі зразки візуального мистецтва як засобу спілкування з одноплемінниками і майбутніми поколіннями з'явилися ще в епоху палеоліту. Малюнки на стінах печер чітко доносили суть побуту і традицій. Значущість візуалізації підтверджується у різних сферах діяльності людини.

Наприклад, інструкції про евакуацію на випадок пожежі або аварійної посадки. Чи дієвим було б довге полотно тексту з поясненнями, що робити в екстреному випадку? Інфографіка дає зрозумілий покроковий алгоритм всього за декілька секунд [3].

Із розвитком сучасної інтернет-індустрії дедалі частіше виникають питання щодо відображення різних типів файлів. Одними із них є XML та JSON [4].

JSON – формат даних, що використовується для обміну інформацією між комп'ютерами. Він базується на тексті, який може прочитати людина. JSON дає змогу описувати об'єкти та інші структури даних. Цей формат використовується переважно для передавання інформації через мережу, але також і в інших сферах програмування [5].

XML – це стандарт правильної будови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками. Стандарт визначає набір базових синтаксичних правил для побудови мови описання даних шляхом застосування простих тегів [6]. Цей формат достатньо гнучкий для того, аби бути придатним для застосування в різних галузях.

Базові принципи роботи з цими форматами, а також методи зчитування, аналізу, структурування та запису даних в них детально описано у [7].

2. Постановка задачі

Виходячи з проведеного аналізу, актуальною задачею є розробка та реалізація повноцінного програмного продукту, який буде забезпечувати можливість редагувати, створювати, зберігати вміст файлів XML- та JSON-форматів; надаватиме кольорове та графічне розділення елементів програмного тексту. При цьому кількість одночасно відкритих файлів повинна бути до 7; кількість рядків у файлі – – до 1000; швидкість аналізу – 45 рядків/сек.

3. Створення візуального редактора

Провівши аналіз особливостей відображення вмісту для обраних типів файлів, та опираючись на методи зчитування, опрацювання та запису даних [7], розроблена структурна схема програмного засобу.

Вона складається з таких частин (рис. 1):

- підсистема зчитування файлів – відповідає за отримання списку файлів, їх аналіз та генерацію даних;
- підсистема аналізу вмісту – здійснює безпосереднє перетворення елементів файлу та формування списку елементів;
- підсистема відображення – отримує дані зі списку елементів та створює список відповідних їм об'єктів, щоб генерувати результат перетворення.

Така побудова забезпечить найбільш коректну роботу візуального редактора і не призведе до ускладнення та нагромадження коду.

Загальний алгоритм роботи програми, що базується на запропонованій структурі, відображено на рис. 2.

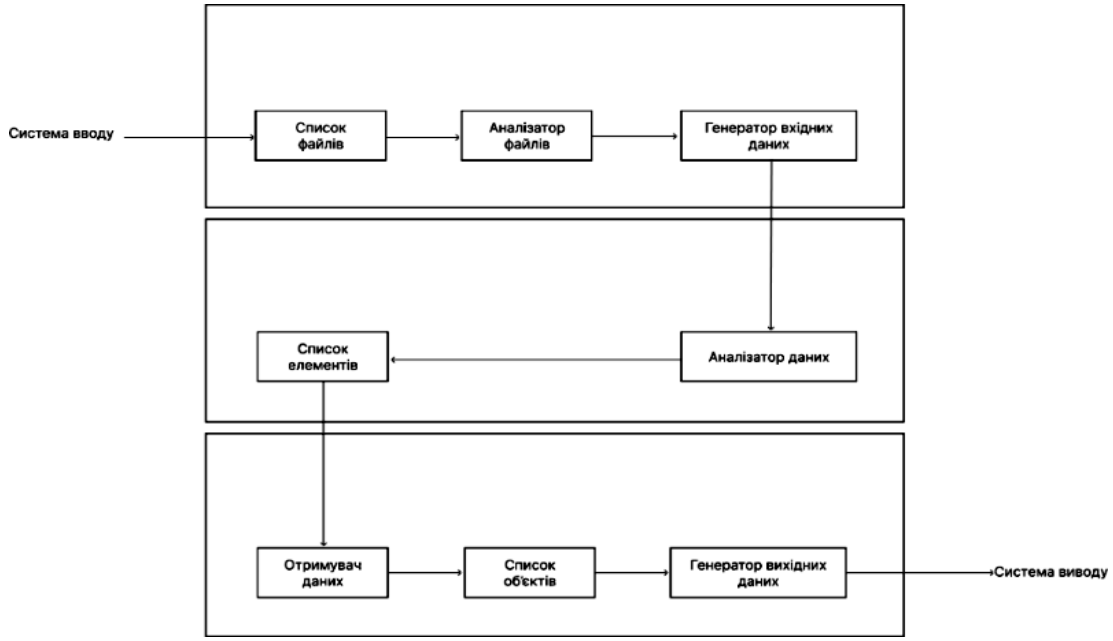


Рис. 1. Структурна схема візуального редактора

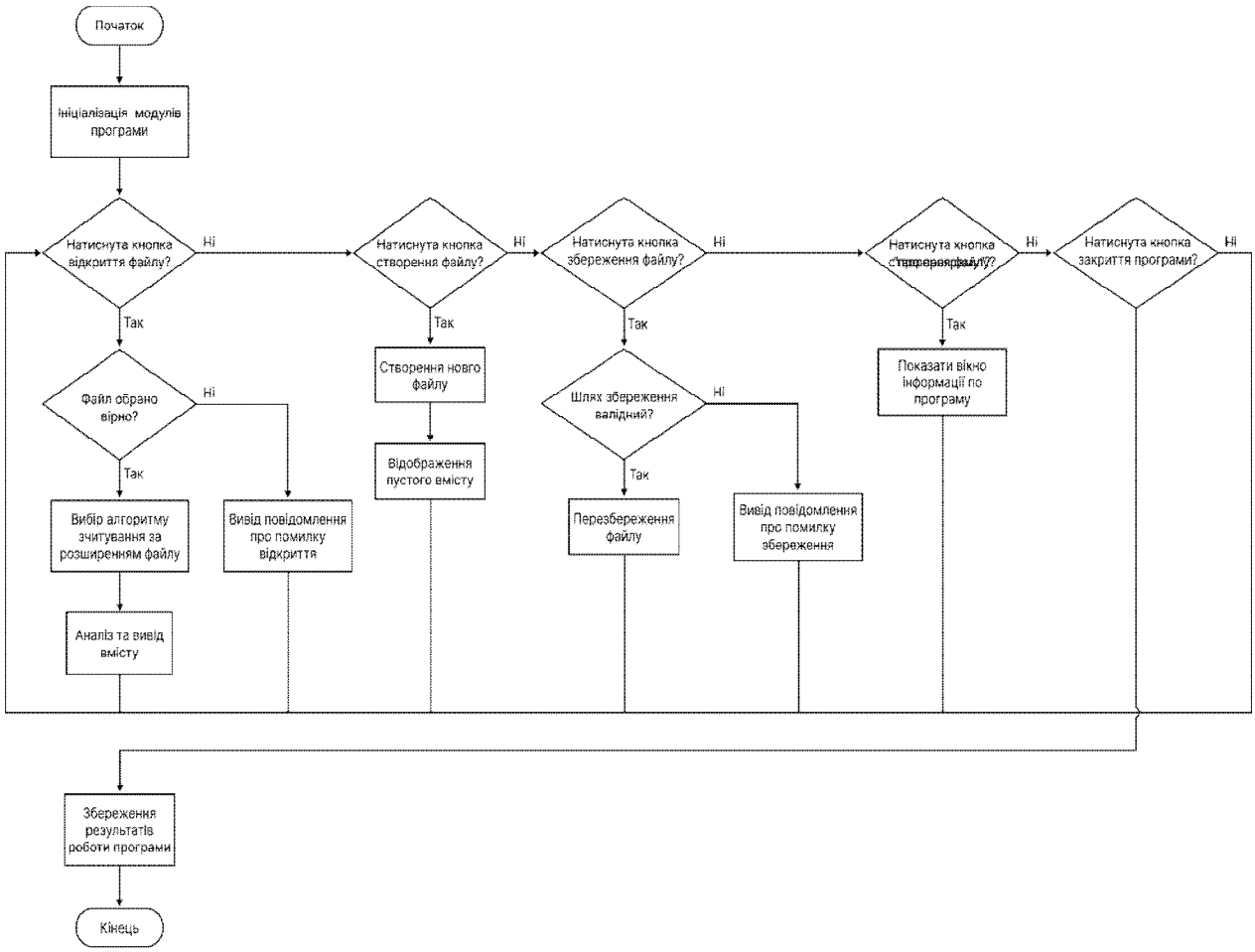


Рис. 2. Схема алгоритму роботи програми

Перед початком роботи із візуальним редактором потрібно ініціалізувати головні елементи, які будуть використовуватись у подальшій роботі з програмою. Після запуску програми користувач має змогу обрати функції, які йому будуть потрібні.

4. Огляд основних функцій програми

1. Відкриття файлу.

Після натискання на відповідну кнопку, з'являється стандартне вікно, яке дає змогу обрати файл. Коли файл обраний, зберігається повний шлях до нього, оскільки ці дані згодом будуть проаналізовані та розділені на окремі елементи. Для спрощення пошуку файлів із потрібним форматом було створено обмеження, а саме – програмним чином було встановлено пріоритетні розширення. Це дає змогу уникнути проблем із подальшим аналізом вмісту.

Створена перевірка на відкриття файлу, яка повертає значення типу BOOL. Ця перевірка є обов'язковою, оскільки цей файл може використовуватись іншою програмою, і для того, щоб не виникало проблем у редакторі та у операційній системі, від подальшого виконання алгоритму залежить, які значення буде повернено.

Функція аналізу зчитує вміст та, залежно від розширення, перетворює його у відповідний тип. Особливістю роботи цього алгоритму є можливість рекурсивного виклику. Файли можуть мати у собі багаторівневу структуру, тому розробити універсальний інструмент без рекурсивного виклику неможливо. Візуалізація даних відбувається паралельно з аналізом. Це зроблено для того, щоб уникнути додаткових затрат ресурсів пам'яті на тимчасове зберігання даних. Вся інформація, що була проаналізована, зберігається у батьківському віджеті, що в майбутньому полегшує доступ до неї.

Редагування вмісту відбувається за допомогою базових елементів QLineEdit. У редакторі можна змінювати ключі, значення і додавати нові елементи у внутрішні об'єкти.

Елементи потрібних типів користувач має змогу додавати до об'єктів та масивів, які були попередньо зчитані. Можна додавати прості та складені типи елементів, які будуть додані у кінець обраного об'єкту. Користувачеві потрібно розмістити курсор у межах обраного елементу та натиснути праву кнопку миші. З'являється меню, в якому можна обрати тип (рис. 3).

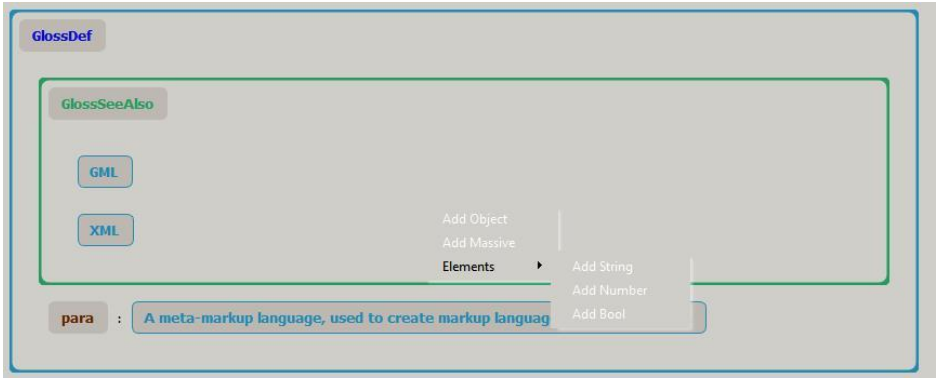


Рис. 3. Додавання нових об'єктів

2. Створення файлу.

Після отримання відповідного сигналу відкривається діалогове вікно, в якому користувач зможе обрати назву, формат та шлях для подальшого створення та збереження. Тут є два поля вводу, в одному з яких користувач вільно може написати назву файлу, а інше закрите від прямого редагування, оскільки у ньому потрібно обрати шлях до папки, де буде створено файл.

Дані, які користувач введе, тимчасово зберігаються у діалоговому вікні, і тільки після натискання кнопки Create вони будуть передані програмі для подальшого аналізу та виведення. Після закриття віджету створюється файл із відповідним шляхом та назвою, та перевіряється доступ на читання, оскільки можливі помилки під час створення.

Залежно від формату файлу, який користувач вибрав, відображенню присвоюються різні параметри головного об'єкту, в який користувач зможе додати елементи (рис. 4).

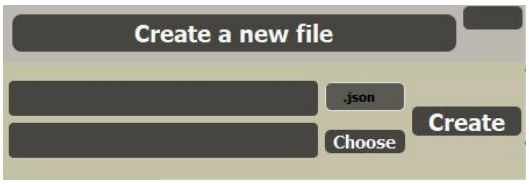


Рис. 4. Вікно створення файлу

3. Збереження файлу.

Після натискання відповідної кнопки відбувається зчитування із активного вікна інформації. Формується пакет даних, який додається у відповідний тимчасовий документ залежно від розширення файлу. Перевіряється наявність файлу у папці, і при негативному результаті виводиться помилка користувачеві, при позитивному – вміст перезаписується.

5. Тестування візуального редактора JSON- та XML-форматів

Тестування проводилося для кожної підсистеми зокрема, оскільки кожний елемент програми є важливим.

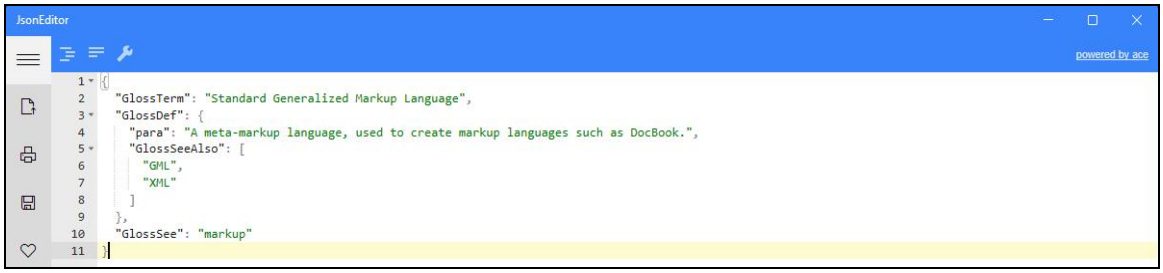
На рис. 5–7 наведено основні візуальні особливості створеного редактора, бо саме вони визначають зручність у користуванні.

Рис. 5. Вивід інформації про відкриті файли

Робота з JSON-форматом:

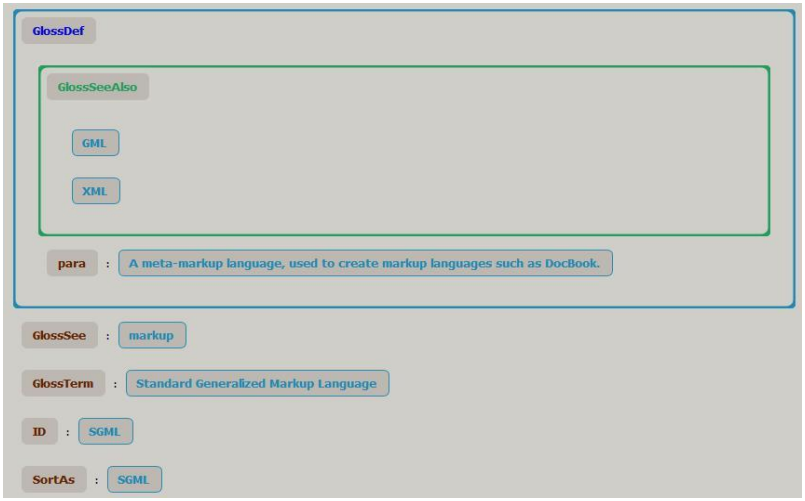
```
{
    "GlossTerm": "Standard Generalized Markup Language",
    "GlossDef":
    {
        "para": "A meta-markup language, used to create markup languages such as DocBook.",
        "GlossSeeAlso": ["GML", "XML"]
    },
    "GlossSee": "markup"
}
```

а



б

Рис. 6. Візуалізація файлу JSON: а – вхідний текст; б – у редакторі Free JSON Editor



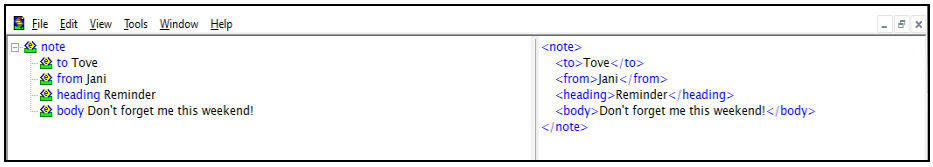
в

Рис. 6 (ghjljd;tyyz). Візуалізація файлу JSON: в – у розробленому редакторі {JX} Editor

Робота з XML форматом:

```
1  <note>
2      <to>Tove</to>
3      <from>Jani</from>
4      <heading>Reminder</heading>
5      <body>Don't forget me this weekend!</body>
6  </note>
```

а



б

в

Рис. 7. Візуалізація файлу XML: а – вхідний текст; б – у редакторі Free XML Editor; в – у розробленому редакторі {JX} Editor

Тестування створення, відкриття та збереження файлів продемонструвало адекватну і коректну роботу створеного редактора. При цьому забезпечено можливість кольорового та графічного розділення елементів програмного тексту.

6. Висновки

Дослідження показало актуальність та доцільність задачі вдосконалення засобів редагування для програмних файлів. Тому реалізовано візуальний редактор XML- та JSON-форматів, що

дозволяє відобразити вміст відповідних файлів у вигляді наглядних блоків. Розділено усі можливі типи даних на різні кольори, щоб користувач зміг легко орієнтуватися у відображеному файлі. Додано можливість редагувати вміст, додаючи елементи різних типів. Розроблено загальний алгоритм роботи редактора та детально розкрито всі його основні функціональні вузли.

Запропонований програмний продукт забезпечує можливість, створювати, редагувати, зберігати вміст файлів XML- і JSON-форматів, та надає кольорове і графічне розділення елементів програмного тексту. Візуальний редактор дозволяє одночасне відкриття 7 файлів розміром до 1000 рядків. При цьому швидкість опрацювання становить близько 45 рядків/сек.

Крім того, створений редактор має якісну перевагу над аналогічними засобами, оскільки в ньому присутнє графічне відображення вмісту.

Отже, можна зазначити, що розроблений візуальний редактор XML- та JSON-форматів має перспективу впровадження та подальшого розширення функціоналу відповідно до вимог ринку та відгуків користувачів.

Список літератури

1. Duncan J. (1984). Selective attention and the organization of visual information. Journal of Experimental Psychology: General, 113(4), 501–517. DOI: 10.1037/0096-3445.113.4.501 G. Wiesen. Visual Editing (Accessed: 14 September 2022).

2. O'Brien C. (2018). Toss Out Goodbye clutter and useless paper – welcome to the digital workplace. / Ciara O'Brien // The Irish Times. URL: (accessed: 15 September 2022).

3. Mudrenko S. Data Visualization. URL: https://mind.ua/openmind/20230899-infografika-dlya-biznesu-yak-vizualizaciya-danih-vplivae-na-prijnyattya-rishen (accessed: 14 September 2022).

4. Wang G. (2011). "Improving Data Transmission in Web Applications via the Translation between XML and JSON", Third International Conference on Communications and Mobile Computing, 2011. Pp. 182–185. DOI: 10.1109/CMC.2011.25 (accessed: 15 September 2022).

5. Abd El-Aziz A. A. and Kannan A. (2014). "JSON encryption", 2014 International Conference on Computer Communication and Informatics. Pp. 1–6. DOI: 10.1109/ICCCI.2014.6921719 (accessed: 15 September 2022).

6. "IEEE Standard for Learning Technology-Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata", in IEEE Std 1484.12.3-2020 (Revision of IEEE Std 1484.12.3-2005).Pp. 1–58, 7 April 2020. DOI: 10.1109/IEEESTD.2020.9059045 (accessed: 20 September 2022).

7. Sukumar P. (2020). How to parse JSON in C++. / Paul Sukumar // LinuxHint. URL: https://linuxhint.com/parse-json-data-cpp/(accessed: 14 September 2022).

PRINCIPLES OF CREATION OF VISUAL EDITOR OF XML AND JSON FORMATS

I. Kutnyk, O. Lashko

Lviv Polytechnic National University,
Computer Engineering Department
Authors' e-mail: *kutnykvana6@gmail.com, lashko.oksana@gmail.com*

This article considers the peculiarities of the perception of large amounts of textual information, and analyzes the needs for visual editing. The implementation of a software product that works with XML and JSON formats and provides graphical and color highlighting of the main elements of the syntax.

The structure of the program was developed and divided into several modules: file reading, content analysis and display. The general algorithm of work of the program is designed. The functionality for opening the file and its further analysis in the program has been created. Visualization of the read and analyzed file is presented.

The aim of the article is to display the results of research on the problem of data visualization in information processing by users, as well as highlight the results of software implementation, which provides editing, creating, saving XML and JSON files and supports color and graphical separation of program text elements. At the same time, up to 7 simultaneously open files with the number of lines in the file – up to 1000 are guaranteed.

Keywords: file reading, content analysis, visual editing, JSON, XML.