XML to JSON Converter API Documentation

Introduction

The XML to JSON Converter API is a web service that allows you to convert XML content to JSON format. This documentation provides details on how to use the API, its endpoints, and expected behavior.

Base URL: it runs at https://localhost:44324/

Endpoints

1. **Convert XML to JSON and store it on file system with given name**

   - **Endpoint:** 'XmlToJson/Convert'
   - **Method:** POST
   - **Description:** Validate input, check for valid XML, convert to JSON, store on file system
   - **Request format:** Multipart form with an XML file and file name
   - **Request parameters:**
     - xmlFile (required) – file to be converted
     - filename (required, letters and digits only) – desired name for the new file
   - **Response format:** JSON
   - **Response status codes :** 200 and 400

Usage

1. **Request**
   POST /XmlToJson/Convert
   Content-Type: multipart/form-data

   xmlFile (required) – file to be converted
   filename (required, letters and digits only) – desired name for the new file

2. **Response**

   Success:

   {"message": "Conversion successful."}

   Error:

   Error with converting

   {"error": "Error converting XML to JSON."}

   Invalid XML error

   {"error": "Invalid XML data."}

   Error writing file on disk

{"error": "Failed to write data on disk."}

General error

{"error": "An unexpected error occurred."}

3. **UI**
    - UI contains form with two input fields and validation for both.
    - Submit button is disabled when no file is uploaded or name is invalid (is empty, contains symbols different from digits or letters).
    - Handling success and error and show them to the client.
4. **Backend**
    - Write successfully converted JSON into directory described in appsettings.json. If directory not exists application creates it.
    - Application runs asynchronously
    - Input is checked at the entry point for existing file and correct name
    - Application uses two custom exceptions for appropriate handling of errors
    - Main logic is stored into interactors for easier testing and separation of responsibilities
5. **Tests**
    - Test project is available. It uses xUnit for unit testing and Moq for mocking objects.
    - Multiple tests are included testing controller and interactors