Stream API - Error Handling

While working with streaming APIs they sometimes return errors due to network interruptions, server-side problems, data transmission, etc. So to handle these errors every API uses their own error-handling mechanisms during the streaming process. It makes the application robust and resilient. So the commonly used error-handling practices are –

Error Event Listeners – Almost all the streaming APIs support error event listeners. Error event listeners come into the picture when an error occurs and allow you to handle the error appropriately. It is can used with suitable objects like WebSocket, Fetch API, or ReadableStream.

Try-Catch Block – You are allowed to use a try-catch block to handle errors while working with synchronous code in a specific type of stream.

Promises and Async/Await – While using Promises or Async/Await with streaming APIs you can use a catch block to handle errors that occur during streaming.

Backoff and Retry Method – If your error is not temporary then you can use the backoff and retry method. In this method, the application waits for the data for a short period of time and if the data is not received in that time period then it retries from the failed operation.

User-friendly error message – If the error occurs, then provide a simple and user-friendly error message to the end user to avoid displaying technical details that may confuse users and can able to avoid security risks.

Data Validation – Always make sure that the incoming data from the streaming API is properly validated and sanitized to avoid data format errors or unexpected data tends to process issues.

Conclusion

Always thoroughly checks the error handling implementation to make sure that it works properly. Now in the next article, we will learn about body data in the fetch API.