Readable Stream | Belajar Membuat Aplikasi Back-End untu..

dicoding.com/academies/261/tutorials/14647

Readable Stream

Pada materi sebelumnya Anda sudah mengetahui cara mengakses berkas melalui fungsi fs.readFile(). Fungsi readFile baik versi asynchronous ataupun synchronous, bekerja dengan membaca berkas hingga selesai sebelum mengembalikan data. Itu berarti bila Anda menggunakannya untuk mengakses berkas yang besar, maka akan membutuhkan waktu lama dan memori yang besar untuk mendapatkan hasilnya. Hal ini sungguh tidak efektif.

Solusinya adalah dengan menggunakan teknik stream. Teknik ini tidak membaca berkas secara sekaligus, tapi dengan mengirim bagian demi bagian. Cara inilah yang digunakan oleh YouTube agar video dapat ditampilkan seketika kepada pengguna.

Teknik stream merupakan salah satu konsep fundamental yang mendukung aplikasi Node.js bekerja. Teknik ini dapat menangani kasus baca tulis berkas, komunikasi jaringan, atau beban kerja apapun agar dapat berjalan dengan lebih efisien. Sabar dulu yah, kasus yang disebutkan tadi terlalu kompleks untuk kita pelajari sekarang. Untuk memahami bagaimana stream bekerja kita gunakan kasus paling sederhana, yakni membaca teks pada berkas secara bagian-per-bagian.

Kita dapat membuat readable stream dengan menggunakan method createReadStream() dari core module fs.

```
const fs = require('fs');
const readableStream = fs.createReadStream('./article.txt', {
    highWaterMark: 10
});
readableStream.on('readable', () => {
    try {
        process.stdout.write(`[${readableStream.read()}]`);
    } catch(error) {
        // catch the error when the chunk cannot be read.
    }
});
readableStream.on('end', () => {
    console.log('Done');
});
```

Fungsi createReadStream() menerima dua argumen. Yang pertama adalah lokasi berkas yang hendak dibaca, dan yang kedua adalah objek konfigurasi. Di dalam objek konfigurasi kita bisa menetapkan ukuran buffer melalui properti highWaterMark. Nilai default dari properti ini adalah 16384 bytes (16kb). Anda tidak perlu menetapkan properti ini bila ingin tetap memiliki nilai default. Namun karena kita hanya menggunakan berkas teks yang ukurannya sangat kecil, jadi kita buat ukuran buffer menjadi 10 bytes. Itu artinya berkas akan dibaca setiap 10 karakter (1 karakter = 1 bytes).

> Buffer di dalam stream adalah memori sementara yang digunakan oleh stream dalam menyimpan data hingga data tersebut dikonsumsi.

createReadStream() mengembalikan EventEmitter, di mana kita dapat menetapkan fungsi listener setiap kali event readable dibangkitkan. Event readable akan dibangkitkan ketika buffer sudah memiliki ukuran sesuai dengan nilai yang ditetapkan pada properti highWaterMark, dalam arti buffer sudah siap dibaca. Kemudian event end akan dibangkitkan setelah proses stream selesai.

Bila kode di atas dijalankan maka akan menghasilkan output seperti ini:

```
1. Stream di Node.js
2. Teknik stream merupakan salah satu konsep fundamental yang mendukung
   aplikasi Node.js bekerja. Teknik ini dapat menangani kasus baca tulis
   berkas, komunikasi jaringan, atau beban kerja apapun agar dapat berjalan
   dengan lebih efisien.
```

ini konten buat note nanti