Calcolo del checksum MD5 di un file con Node.js

Il checksum MD5 è una sequenza di 32 caratteri esadecimali che rappresenta un'impronta digitale univoca del contenuto di un file. È usato comunemente per verificare l'integrità dei file, assicurandosi che non siano stati modificati. In Node.js, calcolare il checksum MD5 di un file è un'operazione semplice grazie al modulo built-in crypto. In questo articolo vedremo come calcolare l'hash MD5 di un file usando Node.js e il modulo crypto.

Per calcolare il checksum MD5, useremo il modulo `crypto` (per generare l'hash) e il modulo `fs` per leggere il file. Entrambi questi moduli sono già inclusi in Node.js, quindi non è necessario installare dipendenze aggiuntive.

Definiamo una funzione chiamata `calculateMD5` che accetterà come parametro il percorso del file da processare. La funzione leggerà il contenuto del file e genererà il checksum MD5.

```
const crypto = require('crypto');
const fs = require('fs');

function calculateMD5(filePath) {
    return new Promise((resolve, reject) => {
        // Creiamo un'istanza di hash MD5
        const hash = crypto.createHash('md5');

        // Creiamo uno stream di lettura del file
        const stream = fs.createReadStream(filePath);

        // Aggiorniamo l'hash con i dati man mano che li leggiamo dallo stream
        stream.on('data', (chunk) => {
            hash.update(chunk);
        });

        // Quando lo stream è terminato, risolviamo la promise con l'hash finale
        stream.on('end', () => {
            const md5Checksum = hash.digest('hex');
            resolve(md5Checksum);
        });

        // Gestiamo gli errori
        stream.on('error', (err) => {
            reject(err);
        });
    });
}
```

Questa funzione utilizza uno stream per leggere il file pezzo per pezzo e aggiorna progressivamente l'hash MD5. Quando la lettura è completa (`stream.on('end')`), la funzione restituisce il checksum in

formato esadecimale.

In dettaglio:

1. **Creazione dell'istanza di hash**: `crypto.createHash('md5')` crea un oggetto hash MD5.
2. **Lettura del file tramite stream**: `fs.createReadStream(filePath)` legge il file senza caricare tutto in memoria, rendendo il processo più efficiente.
3. **Calcolo dell'hash**: ogni volta che riceviamo un chunk di dati dal file, aggiorniamo l'hash con `hash.update(chunk)`.
4. **Finalizzazione dell'hash**: quando la lettura è completata, chiamiamo `hash.digest('hex')` per ottenere il valore MD5 in formato esadecimale.

Conclusione

Questo metodo è efficiente anche per file di grandi dimensioni, poiché legge i dati in modo incrementale senza dover caricare l'intero file in memoria. Utilizzando questo approccio con Node.js, possiamo calcolare velocemente il checksum MD5 di un file per verificarne l'integrità.

Applicazioni Correlate

-

  ### Node.js Placeholder Image

  Applicazione per la generazione con Node.js di immagini segnaposto.
  DockerDocker ComposeNode.jsJavaScriptExpressJS

Node.js URL Shortener

Implementazione in Node.js di un sistema per l'abbreviazione degli URL.
DockerDocker ComposeNode.jsJavaScriptExpressJSMongoDB

- ### JavaScript App Hash Change

  Applicazione che sfrutta gli hash degli URL per gestire contenuto dinamico in JavaScript. DockerDocker ComposeNode.jsJavaScriptExpressJSMySQL