Module Fs dan HTTP, Async dan Sync, Request dan Response

Sio Jurnalis Pipin, S.Kom.

STMIK - STIE MIKROSKIL
PRODI. TEKNIK INFORMATIKA (S-1)

File System (fs Module)

- Bagian dari Node.js dengan memanggil

  const fs = require('fs')

- Berguna untuk mengakses dan berinteraksi dengan sistem file.
- **Module fs** menggunakan metode **asynchronous** secara default, tetapi juga dapat bekerja secara **sinkron** dengan menambahkan **Sync.**

File System (fs Module)

Method	Fungsi
fs.appendFile()	Menambahkan data ke file. Jika file tidak ada, maka file akan dibuat.
fs.chmod()	Mengubah izin file yang ditentukan oleh nama file
fs.chown()	Mengubah pemilik dan grup file yang ditentukan oleh nama file yang dikirimkan
fs.close()	Menutup file
fs.copyFile()	Menyalin file
fs.createReadStream()	Membuat aliran file yang dapat dibaca
fs.createWriteStream()	Membuat aliran file yang dapat ditulis
fs.mkdir()	Membuat folder baru
fs.mkdtemp()	Buat direktori sementara
fs.open()	atur mode file / buka file
fs.readdir()	Membaca isi direktori
fs.access()	Memeriksa apakah file tersebut ada dan NodeJs dapat mengaksesnya dengan izin

File System (fs Module)

Method	Fungsi
fs.readFile() / fs.read()	Membaca isi file
fs.rename()	Ganti nama file atau folder
fs.rmdir()	Hapus folder
fs.unwatchFile()	Berhenti mengawasi perubahan pada file
fs.watchFile()	Mulai perhatikan perubahan pada file
fs.writeFile() / fs.write()	menulis data ke file

File System – readFile()

- Membaca isi file
- **Implementasi** pada module http
  1. Terima request dari user
  2. **Baca file HTML** pada server
  3. Kirim respon ke client

File System – readFile()

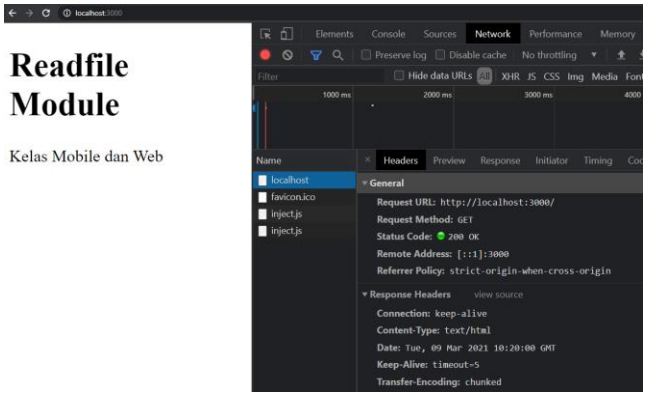
- Buat dalam **folder M02** sebuah file **index.js** dan **index.html**; file html silahkan buat dengan menerapkan **tag h1** dan **p** isi index.js seperti pada gambar berikut:

index.js | index.html

M02-fsModuleAsync > index.js > ...

```
1  const http = require("http");
2  const fs = require("fs");
3
4  //buat object server dengan listen port 3000
5  http
6    .createServer(function (req, res) {
7      //menggunakan modul readFile
8      fs.readFile("index.html", (err, data) => {
9        //mengembalikan pesan error ketika gagal baca file
10       if (err) throw err;
11
12       //http Header
13       res.writeHead(200, { "Content-Type": "text/html" });
14
15       //Respon ke browser / client berupa data dari file index.html
16       res.write(data);
17
18       //Respon diakhiri
19       res.end();
20     });
21   })
22   .listen(3000);
23
```

- Run server pada terminal: **node index.js** dalam folder M02.
- Akses pada localhost:3000



File System – Membuat file

Method yang dapat digunakan untuk membuat file:

- **fs.appendFile()** untuk membuat dan mengisi file;
- **fs.open()** untuk membuat, membuka, dan menulis file;
- **fs.writeFile()** untuk membuat dan menulis file.

File System – appendFile

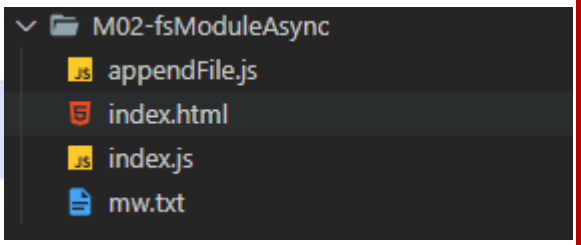
Digunakan untuk membuat dan mengisi file.

- Buat file **appendFile.js**
- Run dengan **node appendFile.js**
- File **mw.text** akan bertambah pada folder **M02**

appendFile.js

M02-fsModuleAsync > appendFile.js > ...

```
1  const fs = require("fs");
2
3  //buat file mw.text dan mengisi dengan data:
4  fs.appendFile("mw.txt", "Kelas Mobile dan Web!", function (err) {
5    if (err) throw err;
6    console.log("Berhasil disimpan!");
7  });
8
9  //run dengan: node appendFile.js
10
```



File System – fs.open()

- Digunakan untuk membuka dan menulis file.

  fs.open(filename, flags, callback)

- Terdapat 3 parameter
  a. Nama file;
  b. Flag;
  c. callback.

File System – fs.open()

Flag	Description
r	To open file to read and throws exception if file doesn't exists.
r+	Open file to read and write. Throws exception if file doesn't exists.
rs+	Open file in synchronous mode to read and write.
w	Open file for writing. File is created if it doesn't exists.
wx	It is same as 'w' but fails if path exists.
w+	Open file to read and write. File is created if it doesn't exists.
wx+	It is same as 'w+' but fails if path exists.
a	Open file to append. File is created if it doesn't exists.
ax	It is same as 'a' but fails if path exists.
a+	Open file for reading and appending. File is created if it doesn't exists.
ax+	It is same as 'a+' but fails if path exists.

File System – fs.open()

- Buat file openwriteFile.js, isi seperti gambar di bawah
- Run dari terminal dengan node operwriteFile.js

openwriteFile.js

M02-fsModuleAsync > openwriteFile.js > ...

```
1  var fs = require("fs");
2
3  fs.open("datamw.txt", "w+", function (err, file) {
4    if (err) throw err;
5
6    // data yang akan kita tulis ke file
7    let data = "Kelas Mobile dan Web!";
8
9    // tulis konten ke file
10   fs.writeFile(file, data, (err) => {
11     if (err) throw err;
12     console.log("Tersimpan!");
13   });
14
15   // baca file
16   fs.readFile(file, (err, data) => {
17     if (err) throw err;
18     console.log(data.toString("utf8"));
19   });
20 });
21
```

Pada fungsi fs.readFile() , digunakan fungsi **toString('utf8')** untuk mengubah **buffer** menjadi teks dengan **encode UTF-8.**

Async dan Sync

- File System defaultnya adalah asynchronouse namun dapat bekerja secara sinkron dengan menambahkan Sync
  - fs.rename()
  - **fs.renameSync()**
  - fs.write()
  - **fs.writeSync()**
- Synchronous adalah sebuah operasi yang akan dijalankan setelah operasi sebelumnya selesai dijalankan atau **berurutan**.
- Asynchronous sebaliknya, tidak perlu menunggu operasi sebelumnya selesai untuk mengeksekusi operasi setelahnya.

Async dan Sync

- Perubahan terjadi pada bentu sync yaitu dapat code dibungkus pada blok try/catch.
- Buat file **mwsore.json**
- Buat dan isi file **asyncRename.js**
- Run dengan terminal: **node asyncRename.js**

asyncRename.js

M02-fsModuleAsync > asyncRename.js > ...

```
1  const fs = require("fs");
2
3  fs.rename("mwpagi.json", "mwsore.json", (err) => {
4    if (err) {
5      return console.error(err);
6    }
7
8    //done
9    console.log("Berhasil mengganti nama!");
10 });
11
```

Async dan Sync

- Buat dan isi file **syncRename.js**
- Run dengan terminal: **node syncRename.js**

- Perbedaan utama di sini adalah eksekusi skrip akan diblok, hingga operasi file berhasil.

Request dan Response

- Pada http server, bentuk req dan respon dapat dilihat seperti ini **http.createServer(function (req, res) {})**
- Pada aplikasi ExpressJs bentuknya ada

  app.get('/', function (req, res) {})
- **Request** merupakan permintaan HTTP berisi permintaan permintaan parameter string, konten, atribut HTTP header.
- **Response** merupakan respon HTTP, yaitu ketika menerima permintaan dikirim ke data respon klien.

HTTP Client

- Digunakan dalam pembuatan HTTP Request dan pembacaan HTTP Response.
- Mengambil data dari server lain untuk diproses lebih lanjut dari sisi server
- Http client pada NodeJS yaitu **http.request** dengan parameter options, dan callback.
- Property dalam option yaitu:
  - **host**, nama domain atau IP server tujuan. Nilai standar adalah “localhost”.
  - **hostname**, sama seperti host. Gunakan nilai ini untuk mendukung url.parse.
  - **port**, port dari server yang dituju. Nilai standar adalah 80.
  - **localAddress**, antarmuka lokal yang ingin diikatkan pada koneksi lokal.
  - **socketPath**, Unix Domain Socket (gunakan salah satu dari host:port atau socketPath).
  - **method**, method dari HTTP Request. Nilai standar adalah GET.
  - **path**, path dari request. Nilai standar adalah /. Jika ada, query string juga diikutkan, misalnya: index.html?data=123. Jika terdapat karakter ilegal sebuah Exception akan dilemparkan. Untuk sekarang, karakter ilegal hanya spasi.
  - **headers**, sebuah objek yang berisi header HTTP.
  - **auth**, informasi autentikasi dengan HTTP Basic Authentication (user:pass) jika server memerlukan informasi ini.
  - **agent**, digunakan untuk mengatur HTTP Agent. Pembahasan HTTP Agent akan dilakukan pada bagian lain.
  - **keepAlive**, sebuah boolean untuk menentukan apakah koneksi tetap disimpan untuk request selanjutnya. Nilai standar adalah false.
  - **keepAliveMsecs**, menentukan waktu koneksi akan tetap dibuka, dalam ukuran milisecond. Nilai standar adalah 1000. Property ini hanya relevan jika keepAlive bernilai true.

HTTP Client

- Buat file httpClient.js pada folder M02 lalu run terminal dengan **node httpClient.js**.