NODEJS STANDARD LIBRARY

MODULO FS

► FS è il modulo della libreria standard di NodeJS che permette di interagire con il file system (file e directory). E' agnostico rispetto al file system, quindi funziona allo stesso modo con tutti i file system più utilizzati.

- ► NTFS
- ► FAT32
- ext4
- Permette di creare, cancellare, modificare e interrogare file o directory.

- Tutti i metodi esposti da FS esistono in doppia versione: sincrona e asincrona:
 - e.g. readFileSync() e readFile()
- ▶ Gli argomenti sono gli stessi ma cambia il modo di fruizione del risultato: flusso standard del codice nel primo caso, tramite callback nel secondo.

Importiamo il modulo fs

JS

const fs = require('fs');



import * as fs from 'fs';

import {readFileSync} from 'fs';

fs.open() e fs.close()

- ► Input:
 - path String -> Path al file da aprire
 - flags (opzionale)
 String -> Flags di lettura/scrittura (r/rw)
- Restituisce
 - ► File descriptor richiesto o un errore Il file descriptor è un numero intero che rappresenta il file all'interno del processo. Va usato in tutte le successive operazioni che lo richiedono come parametro. E' importante chiuderlo una volta terminate le operazioni.

- Input:
 - ► fd Integer -> File descriptor da chiudere
- Restituisce
 - undefined

fs.appendFile()

► Input:

path
String -> Path al file da aprire

String | Buffer -> Dati da inserire nel file

options (opzionale) Object -> Encoding e flag

Restituisce

undefined

fs.copyFile()

► Input:

String -> Path al file da copiare

String -> Path del file creato

► flags (opzionale) String -> flag di copia

- Restituisce
 - undefined

fs.mkdir()

Input:

path

options

Restituisce

undefined

String -> Path della cartella da creare

Object -> Recursive true/false

fs.rmdir()

- ► Input:
 - path
 - options
- Restituisce
 - undefined

- String -> Path della cartella da rimuovere
- Object -> Recursive true/false (experimental)

fs.readdir()

► Input:

path
String -> Path della cartella di cui leggere il contenuto

options
Object -> Encoding e withFileTypes

Restituisce

Un array con il contenuto della cartella

fs.readFile() e fs.read()

- ► Input:
 - path String -> Path al file da leggere
 - options (opzionale)Object -> Encoding e flag
- Restituisce
 - Viene restituito il contenuto del file come Buffer nel caso di encoding non specificato, o come stringa nel caso di encoding soecificato.

- ► Input:
 - ► fd Integer -> File descriptor da leggere
 - options (opzionale)Object
- Restituisce
 - bytesRead + buffer bytesRead è un intero che rappresenta il numero di byte letti. All'interno di buffer sono memorizzati tutti i byte letti. Per ottenere una stringa è necessaria una conversione esplicita.

fs.writeFile() e fs.write()

- Input:
 - path
 String -> Path al file su cui scrivere
 - dataString | Buffer -> Encoding e flag
 - options (opzionale)Object -> Encoding, mode e flag
- Restituisce
 - undefined

- Input:
 - ► fd Integer -> File descriptor su cui scrivere
 - string
 String -> Stringa da scrivere sul file
 - position (opzionale) Integer -> Offset all'interno del file dove scrivere
 - encoding (opzionale)String -> Codifica del parametro string
- Restituisce
 - undefined

fs.rename()

Input:

oldPath

newPath

Restituisce

undefined

String -> Path del file da rinominare

String -> Nuovo path del file

fs.unlink()

- Input:
 - path
- Restituisce
 - undefined

String -> Path del file da cancellare

fs.stat()

- ► Input:
 - path
 String -> Path del file o cartella di cui estrarre informazioni
- Restituisce
 - ▶ stat

Un oggetto di tipo fs.Stat che racchiude tutte le informazioni del file o della cartella, come dimensioni, data di creazione, data di ultima modifica, permessi etc.