Ottenere informazioni sui file in C

Ottenere informazioni su un file in C può essere essenziale per capire la natura del file stesso e per gestire correttamente le operazioni di input/output (I/O). In C, la libreria standard fornisce funzioni che permettono di accedere a informazioni importanti sui file, come la loro dimensione, permessi, data di creazione e modifica, e altri attributi. Di seguito, esploreremo come ottenere questi dettagli con esempi pratici e spiegazioni.

Quando lavoriamo con i file, i metadati sono informazioni che descrivono il file stesso, come:

- Dimensione del file
- Tipo di file (ad esempio, file regolare o directory)
- Permessi di accesso (lettura, scrittura, esecuzione)
- Data di creazione, modifica, e accesso
- ID del proprietario e del gruppo

In C, la libreria standard e POSIX (su sistemi Unix-like) forniscono strumenti e funzioni per accedere a questi dati. Due delle funzioni più usate per ottenere informazioni sui file sono `stat` e `fstat`.

La funzione `stat` è una delle funzioni più comuni per ottenere informazioni sui file. Si trova nella libreria `<sys/stat.h>` ed è supportata su sistemi POSIX (come Linux e macOS). Questa funzione prende in input il percorso del file e riempie una struttura di tipo `struct stat` con i metadati.

Ecco la sintassi della funzione `stat`:

```
#include <sys/stat.h>
int stat(const char *pathname, struct stat *buf);
```

- `pathname`: percorso del file di cui si vogliono ottenere informazioni.
- `buf`: puntatore a una struttura `stat` che verrà riempita con le informazioni sul file.

Se la funzione ha successo, restituisce `0`. In caso di errore, restituisce `-1`.

La struttura `stat` contiene diversi campi che descrivono il file:

```
struct stat {
    dev_t     st_dev;     /* ID del dispositivo
contenente il file */
    ino_t     st_ino;     /* Numero dell'inode */
    mode_t    st_mode;    /* Tipo di file e permessi
*/
    nlink_t   st_nlink;   /* Numero di link */
    uid_t     st_uid;     /* ID dell'utente
proprietario */
    gid_t     st_gid;     /* ID del gruppo
proprietario */
    dev_t     st_rdev;    /* Tipo di dispositivo (se
è un dispositivo speciale) */
    off_t     st_size;    /* Dimensione totale, in
byte */
    blksize_t st_blksize; /* Dimensione del blocco di
I/O */
    blkcnt_t  st_blocks;  /* Numero di blocchi
allocati */
    time_t    st_atime;   /* Tempo dell'ultimo
accesso */
    time_t    st_mtime;   /* Tempo dell'ultima
modifica */
    time_t    st_ctime;   /* Tempo dell'ultimo cambio
```

```
di stato */
};
```

Vediamo un esempio di codice in cui usiamo `stat` per ottenere informazioni su un file e stamparle.

```
#include <stdio.h>
#include <sys/stat.h>
#include <time.h>

int main() {
    struct stat fileStat;

    // Percorso del file di cui ottenere le
informazioni
    if (stat("file.txt", &fileStat) < 0) {
        perror("Errore nell'ottenere informazioni sul
file");
        return 1;
    }

    printf("Dimensione del file: %lld bytes\n", (long
long) fileStat.st_size);
    printf("Ultimo accesso: %s",
ctime(&fileStat.st_atime));
    printf("Ultima modifica: %s",
ctime(&fileStat.st_mtime));
    printf("Permessi: %o\n", fileStat.st_mode &
0777);
    printf("Numero di link: %ld\n", (long)
fileStat.st_nlink);
```

```
    return 0;
}
```

Questo programma:

- Usa `stat` per ottenere le informazioni su un file chiamato `file.txt`.
- Stampa la dimensione del file, il tempo dell'ultimo accesso e dell'ultima modifica.
- Stampa i permessi e il numero di link.

Il campo `st_mode` contiene i permessi del file codificati in un valore ottale. Per interpretare questi permessi, possiamo fare un'operazione bitwise e con `0777`, ottenendo i permessi in formato `rwxrwxrwx`, dove:

- `r` indica la possibilità di lettura (read),
- `w` la possibilità di scrittura (write),
- `x` la possibilità di esecuzione (execute).

`fstat` è simile a `stat`, ma prende in input un file descriptor anziché un percorso di file. È utile se il file è già stato aperto con `open`. La sintassi è:

```
#include <sys/stat.h>
int fstat(int fd, struct stat *buf);
```

`lstat` è simile a `stat`, ma non segue i collegamenti simbolici. Se si usa `stat` su un collegamento simbolico, verranno fornite le informazioni sul file a cui punta il collegamento. Con `lstat`, invece, si ottengono le informazioni sul collegamento stesso.

Conclusione

In C, ottenere informazioni su un file è abbastanza semplice grazie alle funzioni `stat`, `fstat`, e `lstat`. Queste funzioni permettono di accedere a dettagli importanti sui file, come dimensione, permessi, e tempi di modifica, che possono essere utili per prendere decisioni sulle operazioni I/O.