

オペレーティングシステムの機能を使ってみよう 第5章 ファイル操作システムコール

ファイル操作システムコール

- ユーティリティコマンド (ln, rm, mkdir, rmdir, chmod …) 等が使用しているシステムコール。
(コマンドの一覧は「0510_UNIX コマンド.pdf」を参照)
- この章では主要な7種類だけ紹介する。
- 以下の内容は macOS 10.13 を基準にしているが、一部では分かり易さのために単純化して説明している場合もある。

unlink システムコール

- ファイル（リンク）を削除するシステムコール。
- ディレクトリの削除には使用できない。
- rm コマンドは、このシステムコールを使用。

書式 path 引数でリンクのパスを一つ指定する。

```
#include <unistd.h>
int unlink(char *path);
```

解説 unistd.h をインクルードする。
正常時は 0, エラー発生時は -1 を返す。
エラー原因は perror() 関数で表示できる。

使用例 "a.txt" ファイルを削除する例を示す。

```
if (unlink("a.txt") < 0) { // "a.txt" 削除
    perror("a.txt");      // エラー原因表示
    exit(1);              // エラー終了
}
```

mkdir システムコール

- ディレクトリ（フォルダ）作成するシステムコール。
- mkdir コマンドは、このシステムコールを使用。

書式 path, mode でパスと保護モードを指定する。

```
#include <sys/stat.h>
int mkdir(char *path, int mode);
```

解説 sys/stat.h をインクルードする必要がある。
正常時は 0, エラー発生時は -1 を返す。
エラー原因は perror() 関数で表示できる。
パスに含まれる途中のディレクトリは作らない。

使用例 "newdir" ディレクトリを rwxr-xr-x で作成する例。

```
if (mkdir("newdir", 0755) < 0) { // "newdir" 作成
    perror("newdir");          // エラー原因
    exit(1);                    // エラー終了
}
```

rmdir システムコール

- ディレクトリを削除するシステムコールである。
- rmdir コマンドは、このシステムコールを利用。
- 空でないディレクトリを削除することはできない。

書式 path 引数でディレクトリのパスを一つ指定する。

```
#include <unistd.h>
int rmdir(char *path);
```

解説 unistd.h をインクルードする必要がある。
正常時は 0, エラー発生時は -1 を返す。
エラー原因は perror() 関数で表示できる。

使用例 "newdir" と名付けられたディレクトリを削除する例。

```
if (rmdir("newdir") < 0) { // "newdir" 削除
    perror("newdir");      // エラー原因表示
    exit(1);              // エラー終了
}
```

link システムコール (1/2)

- リンク（ハードリンク）を作るシステムコール。
- ln コマンドは、このシステムコールを利用。

書式 存在するパスと新しいパスを指定する。

```
#include <unistd.h>
int link(char *oldpath, char *newpath);
```

解説 unistd.h をインクルードする必要がある。
正常時は 0, エラー発生時は -1 を返す。
エラー原因は perror() 関数で表示できる。

使用例 1 ファイルにリンク "b.txt" を追加する例。

```
if (link("a.txt", "b.txt") < 0) { // "b.txt" 作成
    perror("link");              // 原因は？
    exit(1);                      // エラー終了
}
```

link システムコール (2/2)

使用例 2 ファイルの移動に応用した例.

```
unlink("b.txt");           // 念のため
if (link("a.txt", "b.txt")<0) { // リンクを作る
    ... エラー処理 ...
}
if (unlink("a.txt")<0) {     // リンクを消す
    ... エラー処理 ...
}
```

オペレーティングシステムの機能を使ってみよ

7/11

symlink システムコール

- シンボリックリンクを作るシステムコール.
- `ln -s` コマンドは、このシステムコールを利用.

書式 シンボリックリンク自身のパスと内容を指定する.

```
#include <unistd.h>
int symlink(char *path1, char *path2);
```

解説 `unistd.h` をインクルードする必要がある.
正常時は 0, エラー発生時は -1 を返す.
エラー原因は `perror()` 関数で表示できる.

使用例 内容が "a.txt" のシンボリックリンク "b.txt" を作る.

```
if (symlink("a.txt", "b.txt")<0) { // リンクを作る
    perror("b.txt");               // エラー表示
    exit(1);                       // エラー終了
}
```

オペレーティングシステムの機能を使ってみよ

8/11

rename システムコール

- ファイルの移動 (ファイル名の変更) を行うシステムコール.
- `mv` コマンドは、このシステムコールを利用.

書式 新旧二つのパスを指定する.

```
#include <stdio.h>
int rename(char *from, char *to);
```

解説 `stdio.h` をインクルードする必要がある.
正常時は 0, エラー発生時は -1 を返す.
エラー原因は `perror()` 関数で表示できる.

引数 `from` は古いパス `to` は移動後の新しいパス.

使用例 "a.txt" のパスを "b.txt" に変更する例.

```
if (rename("a.txt", "b.txt")<0) { // パス名を変更
    perror("rename");             // エラー表示
    exit(1);                      // エラー終了
}
```

オペレーティングシステムの機能を使ってみよ

9/11

chmod (lchmod) システムコール (1/2)

- ファイルの保護モードを変更するシステムコール.
- `chmod` コマンドは、このシステムコールを利用.

書式 パスと保護モード (`rw-rw-rws`) を指定する.

```
#include <sys/stat.h>
int chmod(char *path, int mode);
int lchmod(char *path, int mode);
```

解説 `sys/stat.h` をインクルードする必要がある.
シンボリックリンクが指定された場合, `lchmod` はシンボリックリンクの保護モードを変更する.
正常時は 0, エラー発生時は -1 を返す.
エラー原因は `perror()` 関数で表示できる.

オペレーティングシステムの機能を使ってみよ

10/11

chmod (lchmod) システムコール (2/2)

使用例 "a.txt" の保護モードを "`rw-r--r--`" に変更する.

```
if (chmod("a.txt", 0644)<0) { // "rw-r--r--"
    perror("a.txt");           // エラー表示
    exit(1);                   // エラー終了
}
```

オペレーティングシステムの機能を使ってみよ

11/11