CS422 Lab 2 (Stage 2) PSO

2009.02

Stage 2 of the Lab

1. According to the description for the stage 2, if the `<Document Requested>` in the request is a directory, your HTTP server should return an HTML document with hyperlinks to the contents of the directory.
2. Therefore, we have to be able to tell the difference between the name of a file and the name of a directory.

Functions needed for the stage 2

1. `lstat(const char *path, struct stat *buf)`: Once this function returns, the information about the file or the directory would be stored in the `stat` structure buf.
2. Use `buf.st_mode` as the argument of the macro `S_ISDIR` to tell whether the *path is the name of a file or a directory.

Sample Code

```
if(lstat(fullpath, &statbuf) < 0)
  printf("lstat error!");
if(S_ISDIR(statbuf.st_mode) == 0)
  printf("it’s not a directory.");
```

Functions needed for the stage 2

Once we have found that *path is the name of a directory, we use opendir(...) and readdir(...) to help us traverse the files in that directory.

Functions needed for the stage 2

1. Calculate the number of files/directories in this directory.

Functions needed for the stage 2

1. Calculate the number of files/directories in this directory.
2. Save the related information, such as name, size, and the modified time of these files/directories in a data structure.

Functions needed for the stage 2

1. Calculate the number of files/directories in this directory.
2. Save the related information, such as name, size, and the modified time of these files/directories in a data structure.
3. Sort them according to different comparators (use with `qsort(...)` function provided by C language)

Calculate the number of files/directories

```
num_of_file = 0;
dp = opendir(fullpath);
while((dirp = readdir(dp)) != NULL){
  if(strcmp(dirp->d_name, ".") == 0 ||
  strcmp(dirp->d_name, "..") == 0){
   continue;
  }
  num_of_file++;
}
closedir(dp);
```

Save related information

1. Also, when `readdir(dp)` returns with non `NULL` pointer, the name of the file would be stored in the `d_name` field of the `dir_ent` structure `dp`.

Save related information

1. Also, when readdir(dp) returns with non NULL pointer, the name of the file would be stored in the d_name field of the dir_ent structure dp.
2. Then use lstat(const char *path, struct stat *buf) to read the file or directory, some information like size, and the modified time could be found in the st_size and st_mtime field of stat structure buf.

Save related information

Sometimes we need the time we got to be in readable format, we could use the following code to help us.

```
pmytm = gmtime(&statbuf.st_time);
strftime(mytimebuf, bufsize, "%c", pmytm);
```

Save related information

1. `gmtime(...)` is a function that converts a calendar time into what's called a broken-down time, a `tm` structure.
2. `strftime(...)` then converts the `tm` structure pointed by `pmytm` into readable format, and the result is stored in `mytimebuf` of size `bufsize`.
3. The contents in `mytimebuf` would look like this when using "%c" as the third argument: "`Sat Jan 31 16:45:41 2009`".

Save related information

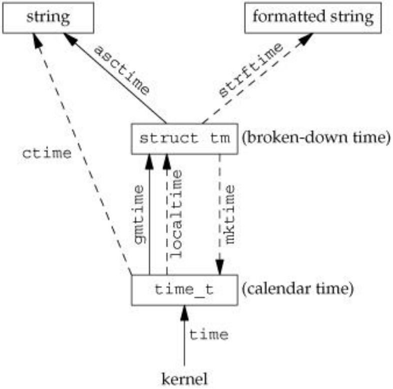

Figure: Relationships of the various time functions

Save related information

A structure to save the related information of a file might look like

```
struct filerec{
  char name[256];
  off_t size;
  time_t mtime;
  time_readable[20];
};
typedef struct filerec Rec;
```

Sorting

We might need to sort an array of filerec sturcture defined as above. We accomplish this using qsort() with appropriate comparators.

An example comparator

Below is the comparator which enables `qsort(...)` to sort the `Rec` array according to their size, in ascending order:

```
int comp_size_asc(const Rec *prec1, const Rec *prec2){
  if (prec1->size < prec2->size)
   return (-1);
  else if (prec1->size == prec2->size)
   return 0;
  else if (prec1->size > prec2->size)
   return (1);
}
```

Using the comparator with qsort(...)

```
qsort(pmyrec, num_of_file, sizeof(Rec),
comp_size_asc);
```

Here pmyrec is a pointer to a Rec array and num_of_file is the size of this array.

1. W. Richard Stevens and Stephan A. Rago, Advanced Programming in the UNIX Environment (2005): Addison-Wesley (6.10 (Time and Date Routines), 4.2 (`lstat` Functions), 4.21 (Reading Directories))