

LIS901b

Lecture 4

the gory details III

Thomas Krichel

2002-03-06

Structure

1. string manipulation (without regular expression)
2. files

Reading

Choi chapter, 10

Choi appendix

simple string manipulation

string `substr(string source, int begin, int end)` returns a substring of *source*, from position *begin*—first position is zero—for *end* chars, the end of the string if *end* is omitted. If *end* is negative, count from the end of the string.

string `trim(string source)` cleans whitespace at the beginning of *source*

int `strlen(string source)` returns the length of the string *source*.

int `printf()` prints a formatted string. string `sprintf()` returns a formatted string.

`sprintf(string format,...)`

where *format* is a string that says how the rest of the arguments are formatted. It starts with a %, then come optional arguments

padding char	default is whitespace, add ' if not zero
alignment	- makes left alignment
minimum width	number, followed by .
precision	number of digits after decimal point

Then comes the required argument, the type specifier

s	string	f	double, using floating point
o	octal integer	e	double, using exponential
d	decimal integer	x	hexadecimal integer
b	binary integer	X	hexadecimal integer
%	the percent as literal		

example

```
<tt><?php
$capitals["New York"] = "Albany";
$capitals["California"] = "Sacramento";
while(list($state,$town) = each ($capitals)) {
    printf ("%'.-20.20s%'.20.20s%s",
           $state, $town, "<br>") ;
}
?></tt>
```

file opening

`int fopen(string filename, string mode)` returns in integer number for the file. Modes are

- `a` append
- `a+` append and read
- `r` read
- `r+` read and write
- `w` write
- `w+` write and read

file reading

`int fpassthru(int file_number)` where *file_number* is the number returned from an earlier opening, shows contents to the output.

`string fread(int file_number, int length)` reads *length* characters from file.

`string fgets(int file_number, int length)` reads a line of up to *length* characters from file.

`array file(int file_number)` reads each line of the file into an array.

`int close(int file_number)` closes file.

file writing and moving about

`int fwrite(int file_number, string text)` writes a string *text* to the file.

`int rewind(int file_number)` rewinds the file.

`int ftell(int file_number)` says where we are in the file.

`int feof(int file_number)` says if we are at the end of the file.

`int fseek(int file_number, int length)` moves to the position *length* in the file.

more functions to deal with files

`int copy(string source, string destination)` copies a file.

`int rm(string source)` removes a file.

`int rename(string old_name, int new_name)` renames a file.

`int file_exists(string file_name)`, `int filesize(string file_name)`, `int filetype(string file_name)`, `int is_dir(string file_name)`, `int is_readable(string file_name)`, `int is_writeable(string file_name)`, `int is_executable(string file_name)`, `int is_file(string file_name)`, and `int is_link(string file_name)` tell us other stuff about a file.

dealing with directories

`int chdir(string directory)` sets the current directory to *directory*

`int opendir(string directory)` opens a directory *directory*

`string readdir(int directory,)` return the next file name in the directory *directory*.

`int rewinddir(int directory)` rewinds a directory *directory*

`int closedir(int directory)` closes a directory *directory*

`int mkdir(string directory, int mode)` makes a directory *directory*, with mode *mode*, set that to 0755.

`int rmdir(string directory)` removes an empty directory *directory*.