# SERVER SIDE PROGRAMMING (PART 4)

## FILE EKSTERNAL (READ, CREATE, WRITE/EDIT)

MATERI KE-10

# PHP MANIPULASI FILE EKSTERNAL

- File handling merupakan proses yang sangat penting dalam aplikasi web
- Proses file handling antara lain: *read, create, edit,* dan *upload*
- File handling harus dilakukan hati-hati. Kesalahan yang umum terjadi: meng-edit file yang salah, mengisi file dengan data yang tidak bermakna (*garbage*), menghapus data atau file secara tidak sengaja

# MANIPULASI FILE

- Membaca File. Ada 2 cara:
  - Fungsi PHP `readfile()`
  - Fungsi PHP `fopen(), fread()`
- Menutup file `fclose()`
- Membaca isi file (baris tunggal) `fgets()`
- Mengecek end-of-file (EOF) `feof()`
- Membaca karakter tunggal `fgetc()`

# FUNGSI `readfile()`

```
CSS = Cascading Style Sheets
HTML = Hyper Text Markup Language
PHP = PHP Hypertext Preprocessor
```

singkatan.txt

```php
<?php
echo readfile("singkatan.txt");
?>
```

baca.php

# `fopen()`

Parameter pertama dari `fopen()` memuat nama dari file, parameter kedua memuat mode yang digunakan untuk membuka file

| Mode | Deskripsi |
|---|---|
| r | **Membuka file untuk dibaca saja**. Pointer file dimulai dari awal file |
| w | **Membuka file untuk ditulis saja**. Menghapus konten dari file atau membuat file baru jika tidak ada di server. |
| a | **Membuka file untuk ditulis saja.** Data yang sudah ada tetap tersimpan. Pointer dimulai dari akhir file. Mmebuat file baru jika file tidak ditemukan di server. |
| x | **Membuka file untuk ditulis saja**. Return FALSE dan error jika file sudah ada di server. |
| r+ | **Membuka file untuk read/write**. Pointer file dimulai dari awal file |
| w+ | **Membuka file untuk read/write**. Menghapus konten dari file atau membuat file baru jika tidak ada di server |
| a+ | **Membuka file untuk read/write**. Data yang sudah ada tetap tersimpan. Pointer dimulai dari akhir file. Mmebuat file baru jika file tidak ditemukan di server. |
| x+ | **Creates a new file for read/write**. Return FALSE dan error jika file sudah ada di server |

# `fread()`

- Fungsi `fread()` untuk membaca isi dari file yang sudah dibuka menggunakan `fopen()`
- Parameter pertama memuat variabel file yang akan dibaca, parameter kedua memuat ukuran bytes maksimum untuk dibaca

Contoh:

```
fread($myfile,filesize("singkatan.txt"));
```

# `fclose()`

- Fungsi `fclose()` digunakan untuk menutup file yang telah dibuka menggunakan `fopen()`
- Penerapan programming yang bagus adalah menutup semua file setelah selesai melakukan manipulasi terhadap file tersebut.
- File yang dibuka menghabiskan *resource* di server dan mencegah proses lain mengakses file tersebut

```php
<?php
$myfile = fopen("singkatan.txt", "r");
// kode untuk memanipulasi file....
fclose($myfile);
?>
```

# CONTOH `fopen(), fread(), fclose()`

```
CSS = Cascading Style Sheets
HTML = Hyper Text Markup Language
PHP = PHP Hypertext Preprocessor
```

singkatan.txt

```php
<?php
$myfile = fopen("singkatan.txt", "r") or die("Tidak bisa membuka file!");
echo fread($myfile,filesize("singkatan.txt"));
fclose($myfile);
?>
```

baca.php

# fgets()

- Fungsi `fgets()` digunakan untuk membaca satu baris dari file

```php
<?php
$myfile = fopen("singkatan.txt", "r") or die("Tidak bisa membuka file!");
echo fgets($myfile);
fclose($myfile);
?>
```

# PHP *CHECK END-OF-FILE* - `feof()`

- Fungsi `feof()` function mengecek apakah pointer telah mencapai "end-of-file" (EOF) / akhir dari file.
- Fungsi `feof()` berguna untuk melakukan perulangan pada data dengan ukuran/panjang yang tidak diketahui.

```php
<?php
$myfile = fopen("singkatan.txt", "r") or die("Tidak bisa membuka file!");
// menampilkan per baris sampai end-of-file / akhir dari file
while(!feof($myfile)) {
  echo fgets($myfile) . "<br>";
}
fclose($myfile);
?>
```

# fgetc()

- Fungsi `fgetc()` digunakan untuk membaca karakter tunggal dalam file
- Contoh: membaca per karakter hinga EOF

```php
<?php
$myfile = fopen("singkatan.txt", "r") or die("Tidak bisa membuka file!");
// menampilkan per baris sampai end-of-file / akhir dari file
while(!feof($myfile)) {
  echo fgetc($myfile);
}
fclose($myfile);
?>
```

# CREATE/WRITE FILE

- Untuk membuat/*create* file menggunakan `fopen()`
- Fungsi `fopen()` digunakan untuk membuka file maupun membuat file bergantung pada mode (lihat table pada slide `fopen()` sebelumnya)
- Jika menggunakan pada file yang belum ada di server, maka akan dibuat file baru untuk menulis/*write* (w) atau untuk menambah/*append* (a)

Contoh:

```
$myfile = fopen("testfile.txt", "w")
```

# fwrite()

- Fungsi `fwrite()` digunakan untuk menulis data ke file
- Parameter pertama memuat variable file yang akan ditulis, parameter kedua adalah *string* yang akan ditulis

```php
<?php
$myfile = fopen("newfile.txt", "w") or die("Tidak bisa membuka file!");
$txt = "Baris satu\n";
fwrite($myfile, $txt);
$txt = "Baris dua\n";
fwrite($myfile, $txt);
fclose($myfile);
?>
```

* Jika file newfile.txt dipanggil lagi dengan menggunakan mode *write* (w), maka data yang baru akan menimpa data yang lama

# MENAMBAHKAN DATA KE FILE YANG TELAH ADA

- Untuk menambahkan data ke file yang sudah ada, gunakan mode *append* (a)

```php
<?php
$myfile = fopen("newfile.txt", "a") or
die("Tidak bisa membuka file!");
$txt = "Baris tiga\n";
fwrite($myfile, $txt);
$txt = "Baris empat\n";
fwrite($myfile, $txt);
fclose($myfile);
?>
```

# BERSAMBUNG

## FILE EKSTERNAL (UPLOAD)