Webmediaproductie 5

PHP - Werken met bestanden

Kristof Michiels

Toepassing: dynamische afbeeldingsgalerij

Een nieuwe toepassing

- Een dynamische webtoepassing met afbeeldingsgalerij
- Deze toepassing zal de mogelijkheid bieden om afbeeldingen toe te voegen via een upload form

Uitgangspunten

- Nieuwe folder met zelfde opbouw als bij de voorbeelden die we eerder zagen
- Kopieer de folders templates en classes
- Maak volgende nieuwe folders aan: css, views, img.
- Zorg voor een vijftal jpg-afbeeldingen in de img folder

De navigatiepagina

- Ook deze toepassing zal uit 2 pagina views bestaan
  - De eerste view toont de galerij
  - De tweede view toont een form die toelaat om nieuwe afbeeldingen toe te voegen

Maak navigatie.php aan in de views folder met volgende code

```
<?php
    return '
    <nav>
        <a href="index.php?pagina=galerij">Galerij</a>
        <a href="index.php?pagina=opladen">Nieuwe afbeelding toevoegen</a>
    </nav> ';
```

Twee pagina views

We maken twee nieuwe pagina's aan in de views folder: galerij.php en opladen.php

```
<?php
    //code voor views/galerij.php
    return '<h1>Afbeeldingsgalerij</h1>';
```

```
<?php
    //code voor views/opladen.php
    return '<h1>Nieuwe afbeelding toevoegen</h1>';
```

De index.php

... zou er ondertussen vertrouwd moeten uitzien

```
<?php
    include_once "classes/Pagina_Data.class.php";
    $paginaData = new Pagina_Data();
    $paginaData->titel = "Dynamische afbeeldingsgalerij";
    $paginaData->content = include_once "views/navigatie.php";
    $userClicked = isset($_GET['pagina']);
    if ($userClicked) {
        $fileToLoad = $_GET['pagina'];
    } else {
        $fileToLoad = "galerij"; }
    $paginaData->content .=include_once "views/$fileToLoad.php";
    $pagina = include_once "templates/pagina.php";
    echo $pagina;
```

Object methods

2 stylesheets toevoegen

- Stel: we willen twee stylesheets toevoegen
- We zouden dit kunnen doen vanuit index.php met volgende code:

```
$paginaData->css = '<link href="css/layout.css" rel="stylesheet">';
$paginaData->css .= '<link href="css/navigatie.css" rel="stylesheet">';
```

- Maar: DRY-principe: don't repeat yourself. Altijd proberen zo weinig mogelijk repetitieve code te schrijven
- We gaan dit oplossen door gebruik van een krachtig concept: object methods

Object method aanmaken

- Object methods zijn net als functies
- Methods worden ook gedeclareerd als functies
- Maar ze worden gedeclareerd binnen een klasse, hier onze Pagina_Data.class.php

```
<?php
    class Pagina_Data {
        public $titel = "";
        public $content = "";
        public $css = "";
        public $embeddedStyle = "";
        public function addCSS($href){
            $this->css .= '<link href="$href" rel="stylesheet">';}
    }
```

$this ?

Wanneer je binnen een klasse definitie verwijst naar het object zelf, dan gebruik je $this

```
public function addCSS($href){
    $this->css .= '<link href="$href" rel="stylesheet">';
}
```

De nieuwe method gebruiken

Je kan nu de nieuwe method gebruiken vanuit index.php

```
$paginaData->addCSS('css/layout.css');
$paginaData->addCSS('css/navigatie.css');
```

Een functie om afbeeldingen te tonen

We passen views/galerij.php aan en starten met volgende code

```
<?php
    return toonAfbeeldingen();
    function toonAfbeeldingen(){
        $out = '<h1>Afbeeldingsgalerij</h1>';
        $out .= '<ul id="afbeeldingen">';
        $out .= '<li>Hier komen de afbeeldingen te staan</li>';
        $out .= '</ul>';
        return $out;
    }
```

Itereren met while loops

While loops herhalen dezelfde blok code zolang aan de voorwaarde is voldaan

```
while ($voorwaarde) {
    //code die dient herhaald te worden
}
```

```
<?php
    $nummer = 1;
    while ($nummer < 5) {
        echo "deze while loop heeft $nummer loops gemaakt<br>";
        $nummer = $nummer + 1;
    }
```

DirectoryIterator

DirectoryIterator

- We kunnen een native PHP object gebruiken om de inhoud van een folder weer te geven: DirectoryIterator
- Hieronder een algemeen voorbeeld waarin DirectoryIterator wordt gebruikt
- Met de valid() method weet je wanneer je te maken hebt met een bestaand item in de folder
- Met de next() method ga je naar het volgende item

```
$filesInFolder = new DirectoryIterator("img");
$numItemsInFolder = 0;
while ($filesInFolder->valid()) {
    $numItemsInFolder = $numItemsInFolder + 1;
    $filesInFolder->next();
}
echo "$numItemsInFolder items gevonden in folder img";
```

views/galerij.php: toonAfbeeldingen() aanpassen

```
function toonAfbeeldingen(){
    $out = '<h1>Afbeeldingsgalerij</h1><ul id="afbeeldingen">';
    $folder = 'img';
    $filesInFolder = new DirectoryIterator($folder);
    while ($filesInFolder->valid()) {
        $file = $filesInFolder->current();
        $filename = $file->getFilename();
        $src = '$folder/$filename';
        $fileInfo = new Finfo(FILEINFO_MIME_TYPE);
        $mimeType = $fileInfo->file($src);
        if ($mimeType === 'image/jpeg') {$out .= '<li><img src="$src"></li>';}
        $filesInFolder->next();
    }
    $out .= '</ul>';
    return $out;
}
```

Afbeeldingen opladen met een form

- We maken een form aan in een nieuw bestand, views/upload-form.php
- We gebruiken een HTML form met method en action eigenschappen
- Als we bestanden willen opladen via HTTP moeten we ook een enctype eigenschap toevoegen en die gelijk stellen aan multipart/form-data
- Ook nieuw: input type="file" zorgt voor een file upload control. Heeft een accept eigenschap

views/upload-form.php

```
<?php
return '
    <h1>Nieuwe JPG afbeeldingen opladen</h1>
    <form method="post" action="index.php?pagina=opladen" enctype="
    multipart/form-data">
    <label>JPG afbeelding vinden om op te laden</label>
    <input type="file" name="image-data" accept="image/jpeg">
    <input type="submit" value="upload" name="new-image"></form>';
```

Deze form tonen

- We passen hiervoor views/opladen.php aan
- Als eerste stap...

```
<?php
    $output = include_once 'views/upload-form.php';
    return $output;
```

$_FILES

$_FILES

- Wanneer we een afbeelding opladen via een HTML form, dan kan de bestandsdata opgevraagd worden via de $_FILES superglobal array
- We gaan via print_r() eerst kijken naar wat PHP ziet
- Pas hiervoor views/opladen.php aan (zie volgende slide)
- Je kan nu de form testen en een jpg bestand selecteren...

$_FILES met print_r()

```php
<?php
//tijdelijke code voor views/opladen.php
$newImageSubmitted = isset($_POST['new-image']);
if ($newImageSubmitted) {
    $output = opladen();
} else {
    $output = include_once "views/upload-form.php";
}
return $output;

function opladen(){
    $out = '<pre>';
    $out .= print_r($_FILES, true);
    $out .= '</pre>';
    return $out;
}
```

$_FILES met print_r(): output

- De array $_FILES heeft 1 index: image-data (= name eigenschap van de upload control)
- Binnen $_FILES['image-data'] zit er nog een array met 5 indices: name, type, tmp_name, error, size
- $_FILES[‘image-data']['tmp_name'] = plaats waar de file data tijdelijk wordt opgeslagen

```
Array (
    [image-data] => Array (
        [name] => afbeelding.jpg
        [type] => image/jpeg
        [tmp_name] => /Applications/XAMPP/xamppfiles/temp/phpYPcBjK
        [error] => 0
        [size] => 119090 )
)
```

Bestanden uploaden met PHP

- Je neemt de tijdelijke filedata en je slaat ze permanent op
- Je moet aangeven in welke folder je wil opslaan en welke filenaam je aan het bestand wil geven
- Er is een native PHP functie die dit voor je doet: move_uploaded_file()
- Eerste argument ($data) moet correcte bestandsdata bevatten
- Tweede argument ($bestemming) moet een bestaande beschrijfbare folder zijn
- Deze functie geeft TRUE terug als het bestand succesvol werd opgeslagen, FALSE bij een fout

```
move_uploaded_file( $data,$bestemming );
```

Een oplader klasse aanmaken

- Bestanden opladen: als PHP ontwikkelaar is het een vaak voorkomende zaak.
- Om code herbruikbaar te maken gaan we een klasse aanmaken
- We maken een nieuw bestand aan, classes/Oplader.class.php

```
<?php
    class Oplader {
        private $filename;
        private $fileData;
        private $destination;
        public function saveIn( $folder ) {
            $this->destination = $folder;
        }
        public function save(){ /*hier nog geen code*/ }
    }
```

__construct

```php
<?php
    class Oplader {
        private $filename;
        private $fileData;
        private $destination;
        //een constructor method aanmaken
        public function __construct($key) {
            $this->filename = $_FILES[$key]['name'];
            $this->fileData = $_FILES[$key]['tmp_name'];
        }
        public function saveIn( $folder ) {
            $this->destination = $folder;
        }
        public function save(){/*hier nog geen code*/ }
    }
```

__construct

- Je kan methods creëren die enkel worden gebruikt bij de creatie van het object
- We noemen dit type method een constructor
- $fileName en $fileData krijgen zo vanuit $_FILES hun invulling bij de creatie van het object
- $key moet hier identiek zijn aan het value attribuut van de upload control (image-data)

Het bestand opslaan

- We moeten nu enkel nog de save-method aanpassen
- We zorgen voor de nodige foutmeldingen indien we niet naar de folder kunnen schrijven...

```
//gedeeltelijke code voor classes/Oplader.class.php
public function save(){
    $folderIsWriteAble = is_writable( $this->destination );
    if( $folderIsWriteAble ){
        $name = "$this->destination/$this->filename";
        $succes = move_uploaded_file( $this->fileData, $name ); }
    else {
        trigger_error("kan niet schrijven naar $this->destination");
        $succes = false; }
    return $succes;
}
```

De oplader klasse gebruiken

- De meeste code zit in onze Oplader klasse
- Aanroepen en gebruik van deze klasse vanuit opladen.php is al wat nodig is om bestanden op te laden

```
function opladen(){
    include_once "classes/Oplader.class.php";
    $oplader = new Oplader("image-data");
    $oplader->saveIn("img");
    $fileUploaded = $oplader->save();
    if ($fileUploaded) {
        $out = "nieuw bestand opgeladen";
    } else {
        $out = "er is iets foutgelopen";
    }
    return $out;
}
```

Img-folder: web server met schrijfrechten

- Vergeet niet de rechten voor everyone op lezen en schrijven te zetten
- Anders kan Apache het opgeladen bestand niet toevoegen aan de img-folder
- Op Mac bvb via rechtsklikken op folder en info selecteren

Opdracht

- Bekijk de broncode grondig die bij deze les hoort
- Zelfstudie: zorg ervoor dat je deze code goed begrijpt tegen morgen

WMP5 - PHP - Werken met bestanden

kristof.michiels01@ap.be