Opzetten van een webapplicatie

- Lezing gaat over het opzetten van een redelijk ingewikkelde web-applicatie.
- Niet ingewikkeld in de zin dat de verwerking zo lastig is
- Ingewikkeld in de zin van
- veel subfuncties
- veel tabellen

Uitdaging:

- hoe organiseer je de code?
- hoe zorg je voor overzichtelijke code?
- hoe zorg je ervoor dat uitbreidingen mogelijk zijn
  en er bij wijzigingen niets op een andere plaats omvalt.

Mijn ervaring met PHP tot 2010

Beperkte ervaring met PHP.

Wel eens wat kleine applicaties gemaakt.

Waarom PHP?

Niet omdat het zo'n fantastische programmeertaal is

- Beschikbaarheid bij hosting providers
- Uitgebreide library
- Goede koppeling met RDBMS-en (MySQL, PostGreSQL, MsSQL, Oracle, Sqllite eva)
- PHP-code kan gemengd worden met HTML
- goede OO-mogelijkheden
- een aantal interessante mogelijkheden die het ontwikkelen vergemakkelijken

Naïeve opzet

- tig PHP-files
- index.php
- subfunc1.php
- subfunc2.php
- etc etc

URL's in de output als

```
http://mijnsite.nl/Applicatie
http://mijnsite.nl/Applicatie/subfunc1.php?...extra parameters
http://mijnsite.nl/Applicatie/subfunc2.php?...extra parameters
```

Opzet van een subfunctie

```
<?php
require("algemeen.inc");

... doe wat gedaan moet worden ....
```

```
?>
<HTML>
... HTML ....
.... evt gemengd met php-code ...
```

Dat kom je voor een wat uitgebreidere applicatie snel van terug.

Gilde Amersfoort

Wat is Gilde Amersfoort?

- oudere vrijwilligers
- Samenspraak
- rondleidingen in Amersfoortse binnenstad en andere wijken, beklimmingen OLV-toren
- 60 gidsen, 15 kantoormedewerkers
- in 2013: plm 1300 rondleidingen, 15.000 gasten
- gereserveerde wandelingen
- aanloopwandelingen

Oude situatie

- doordrukformulieren: derde slecht leesbaar, niet altijd compleet
- indeler moest de formulieren in handen krijgen
- archivering via mappen

Verantwoordelijk bestuurslid voor het kantoor (bij zijn aantreden):

Ik wil zo snel mogelijk van die doordrukformulieren af!

Waarom zelf gemaakt?

prijs!

Voordeel

- goedkoop
- flexibel

Nadeel

CONTINUÏTEIT!

Uitgangspunt lezing

Lezing is gebaseerd op mijn ervaringen bij het bouwen van de applicatie.

Zeker niet het ultieme antwoord!

Fred Brooks (De Magische Manmaand): *There is no silver bullet*

Eerste doelstelling

In eerste instantie elektronische vervanging van het formulier. Al snel uitbreidingen gewenst

Benodigde kennis

- kunnen programmeren
- PHP
- opzetten en kunnen beheren van databases
- weten hoe je die benadert
- kennis van HTML
- begrijpen hoe een webserver en een webapplicatie werken
- evt. kennis van JavaScript
- evt. kennis van OO-programmeren

Algemene werking van een webapplicatie

- gebruiker geeft een URL in
- webapplicatie verwerkt de URL en de meegegeven data
- webapplicatie toont HTML
- gebruiker klikt op een link of
- gebruiker vult een formulier in en klikt op Submit
- en het geheel herhaalt zich

Problemen/uitdagingen

HTTP is stateless: webserver handelt een request af en dat is het

Hoe draag je informatie over van de ene transactie naar de andere?

- GET-parameters via de URL

  ```
  http://...../...?par1=...&par2=....
  ```
- POST-parameters bij formulieren
- PHP-sessies via een cookie

Kenmerk van een administratieve applicatie

Hij is nooit af. Er zijn altijd wel weer dingen die wat anders moeten, er komen nieuwe wensen e.d.

Eerst gekozen opzet

alle URL's hebben de structuur:

```
http://mijnsite.nl/Applicatie/index.php?actie=subfunc1&...extra parameters
```

In index.php krijg je dan constructies als

```
... algemene coding ...
switch($actie) {
case 'subfunc1':
    require("module1.php");
    SubFunc1();
```

```
    break;
case 'subfunc2':
    require("module1.php");
    SubFunc2();
    break;
....
}
```

module1.php

```
<?php

require("database.php");
require("html.php");

function SubFunc1()
{
    ....
    LeesRegelUitDatabase(..);
    ...
    GenereerBeginHTMLPagina();
    ...
    echo <<EOD
<TR><TD>.... $waarde;</TD></TR>
EOD;
    ....
?>
<TR><TD>
.... <?php echo $waarde; ?>
</TD>
</TR>
<?php
}
function SubFunc2()
{
...
}
?>
```

Gerealiseerd in versie 1

- gereserveerde rondleidingen registreren
- rondleidingen op aanloop registreren
- bevestiging naar klant sturen
- mails naar beheerders van speciale objecten
- indeler moet gidsen aan rondleidingen kunnen koppelen
- gidsen moeten informatie over toegewezen rondleidingen krijgen
- informatie voor externe website: komende aanloopwandelingen
- geen rapportages, wel exportmogelijkheden wo naar .xls-files

Nadelen

- onoverzichtelijk wirwar van modules die elkaar aanroepen
- lastig uit te breiden met nieuwe functies
- HTML en code gemengd
- wat moet je doen als je bijv. HTML voor een portable device wilt genereren?

En dus

opnieuw beginnen en versie 2 maken

Versie 2: Wat te doen?

Framework gebruiken
Zend, Cakewalk etc

Voordeel: alles wat je kunt bedenken is aanwezig (mailen, etc)

Nadeel

- steile leercurve
- redelijk star keurslijf

Versie 2: Code en HTML-scheiden

Template engine gebruiken?
Bijv. Smarty
Is in feite een apart taaltje

PHP is dan een betere templatetaal
geen caching...

Programmeren is een complexe bezigheid

- **Alles wat de complexiteit verhoogt:**

- **Alles wat de complexiteit verlaagt:**

Een aantal vuistregels (in volgorde van belangrijkheid)

1. zorg dat het goed draait
2. zorg dat het goed BLIJFT draaien
    - ♦ schrijf zulke code dat een ander begrijpt wat je doet
      die ander kun je zelf wel zijn over een half jaar
    - ♦ schrijf code alsof de volgende die er naar kijkt een psychopaat is die weet waar je woont...
    - ♦ oftewel: **KISS** Keep It Simple, Stupid
3. pas als laatste: zorg dat het een acceptabele performance heeft

MVC-model

Zie Wikipedia

Splits de applicatie in

- models
- views
- controllers
- plus ondersteunende functies
    - ♦ ./lib
      algemene applicatie-onafhankelijke functies
      versturen van mail, lezen van config-files, database-handling
    - ♦ ./helpers
      applicatie-specifieke functies
      formulierafhandeling, HTML-generatie, etc

Models

Models: beelden rijen in database-tabellen en -views af als een object
Models weten niet welke views of controllers er zijn

Er is een ActiveRecord-library voor PHP
Niet gebruikt: vereist PHP 5.3

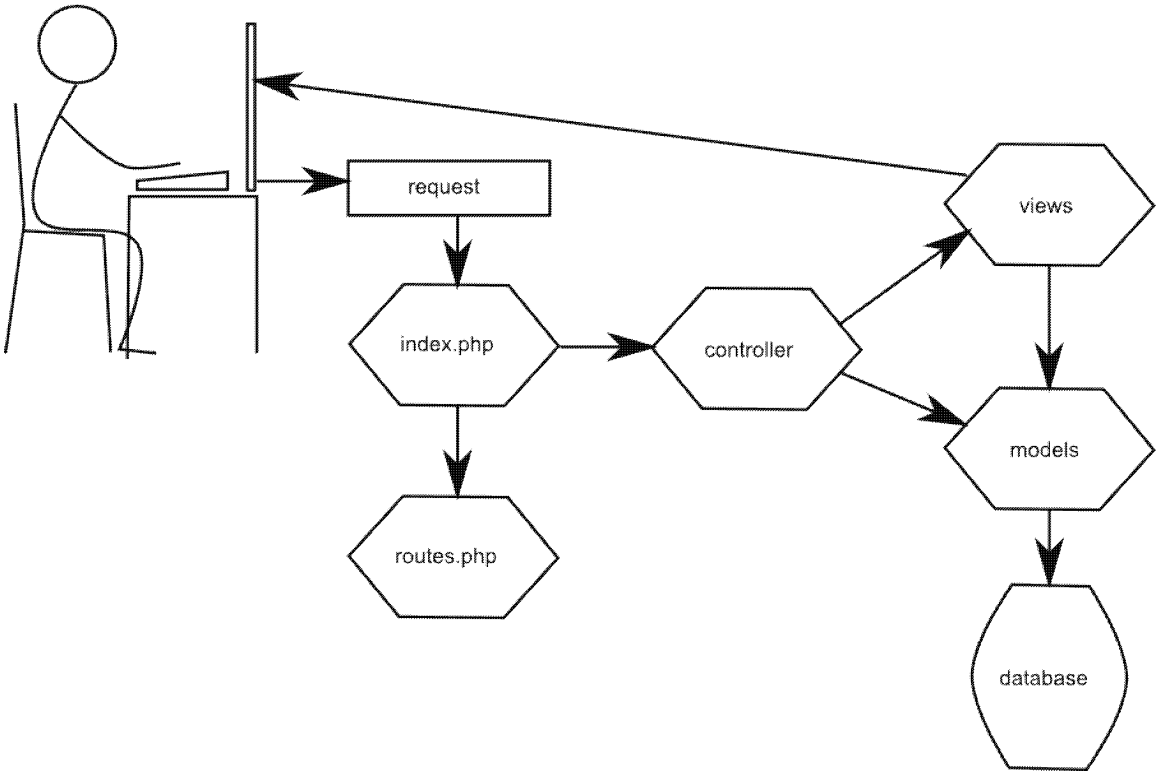
Views

Views genereren de HTML
Views weten welke models er zijn maar niet welke controllers
Zouden in principe andere HTML voor een tablet/smartphone kunnen genereren dan voor een desktop-computer

Controllers

Controllers: bevatten de applicatie-logica.
Hebben kennis van models en views

Flow



Flow

Flow van de afhandeling van een request

- komt altijd binnen bij index.php
- start de sessie
- zet de applicatie op (database-toegang e.d)
- trigger de authenticatie
    - inloggen
    - sessie laten vernieuwen bij langdurige inactiviteit
- maak een RequestObject
    - alle GET- en POST-variabelen
    - config-object
    - user-object

Flow (2)

```
require("routes.php");

$route['hoofdmenu']                 = "controllers/boekingen/hoofdmenu/Hoofdmenu";
$route['....']                      = "controllers/boekingen/.......";
```

zoek de entry op adhv de action-variabele

```
$modulenaam="controllers/boekingen/hoofdmenu.php";
require($modulenaam);
$functienaam="Hoofdmenu";
$functienaam();
```

config-object bevat allerlei parameter-achtige gegevens
config-files zijn PHP-files

Idee: als de applicatie voor een andere organisatie gebruikt gaat worden hoeft de code niet aangepast te worden maar alleen de config-files.
Maar of dat gaat werken??

User-object wordt gevuld adhv de authenticatie. Daarmee kunnen de rechten van een gebruiker bepaald worden.

Controller

```
<?php
defined('_GAEXEC') or die("Restricted access");

function Hoofdmenu()
{
    global $RequestObject;

    .....

    $selector=new BoekingSelector();
    $rows=$selector("datum >= ? AND datum <= ?",
        Array($datumvan,$datumtot));
    foreach($rows AS $row) {
        ... $row->datum ....
        ... $row->tijd ....
    }

    $view=new View("boekingen/hoofdmenu",
```

```
    'arg1',$waarde1,
    'arg2',$waarde2);

    echo StandaardHelper::StartOfPage("Hoofdmenu");
    echo $view->render();
    echo StandaardHelper::EndOfPage();
}
```

Controller (2)

PHP-features

1. __autoload: on-the-fly laden van classes
2. __get en __set: magische getters en setters tbv attributen
3. ob_start cs: gegenereerde output niet naar de output sturen maar opvangen

Door 1) hoef je niet expliciet aan te geven welke PHP-files ge-included moeten worden.

Door 2) zijn constructies mogelijk als

```
    if ($boeking->datum > ...)
       ....
    $boeking->datum='.....';
```

Door 3) kan een view gewoon HTML genereren. ob_start cs wordt gebruikt om deze output af te vangen en terug te geven aan de aanroeper.

Controller (3)

Per tabel/view:

- een ActiveRecord-model
  - ♦ beeldt één enkele rij af
  - ♦ insert, update, delete-methods
- een ActiveRecordSelector
  voor het vinden van een aantal rijen
  Generatie van deze classes via een eenvoudig Perl-script adhv de tabel/view-definitie in de database.

Views

Mengeling van PHP-code en HTML

HTML of embedden of via print/echo-statements
Wordt opgevangen en teruggegeven via ob_start e.d.

Gebruik van alternatieve syntax voor control-structures (geen { en })

```
<?php
if (....) :
    echo(".....");
    ....
elsif (...) :
    ....
    ....
else :
?>
```

```
    <TR>
        <TD>....</TD>
        <TD>....</TD>
    </TR>
<?php
endif;
```

Omvang applicatie

- 27 tabellen
- 11 views op de database
- 48 controllermodules tbv plm. 100 verschillende acties
- 160 views
- 27 library-classes
- 18 helper-classes
- > 31000 sourceregels waarvan > 23000 code-regels

Situatie nu (1)

Huidige versie is najaar 2010 in produktie gegaan

Uitbreidingen

- beheer van de medewerkers
- aanvragen wandelingen door bezoekers publieke website
- realisatie van rondleidingen
- intekeningen op rondleidingen door gidsen en andere wijze van koppeling rondleidingen en gidsen
- wandelkalender en planningsoverzicht
- klantenenquete mbv lime-survey
- allerlei exports

Situatie nu (2)

- koppeling applicatie aan interne website
- idem externe website
- afhandeling wandelingen
- externe partij (VVV) toegang geven
- reservering OLV-toren via externe website

Geen problemen tegengekomen, geen onverwachte effecten

Performance

- geen issue bij bouwen
- applicatie wordt ook niet zwaar gebruikt
- max. aantal concurrent users af en toe meer dan 1

Klein testje

- wat browsen plus een nieuwe boeking maken
- CPU-tijd webserver plm 1 msec
- mysql: plm 350 databaseaccessen

Performance hack

Door de koppeling met de externe server voor de reservering van de OLV-toren werd standaard bij iedere actie van de gebruiker plm 150 KB binnengehaald.
Opgelost via een eenvoudig caching-mechanisme

Fouten e.d.

Wat is er erger dan een fout in de applicatie?

Een fout die niet gemeld wordt!

Fout-afhandeling

Defensief programmeren

```
if (dit kan niet gebeuren)
throw new MyException("Dit kan niet gebeuren");


try {
   ... voer actie uit
}
catch (MyException $except) {
    ... rapporteer MyException ...
}
catch (Exception $except) {
    ... rapporteer Exception ...
}
```

Wat had beter gekund?

- database niet geheel genormaliseerd
- aparte tabellen voor users (van de applicatie) en medewerkers
- wel erg veel velden in boekingsrecord die niet altijd relevant zijn

Debugging

Aparte lib functie Debugger::DebugPrint(...)

Modules beginnen met

```
define('DEBUG_modulenaam', GeneralHelper::GetDebugFlag(__FILE__));
```

In een configfile

```
$config['debugger']['controllers/modulenaam.php'] = true;
```

GeneralHelper::GetDebugFlag kijkt of voor deze file de setting aanwezig is en aan staat

In de coding

```
if (DEBUG_modulenaam) Debugger::DebugPrint(.... gewenste output ...);
```

Genereert output als

```
<!-- DEBUG
.... de output ...
-->
```

Debugging kan zo zonder aanpassingen in de code aan- en uitgezet worden.

Beheer sources

> 200 php-sources

Mijn Linux-desktop is de master

Wijzigingen:

- lokaal aanbrengen en uittesten
- IDE: vim
- gewijzigde sources overzetten naar webserver

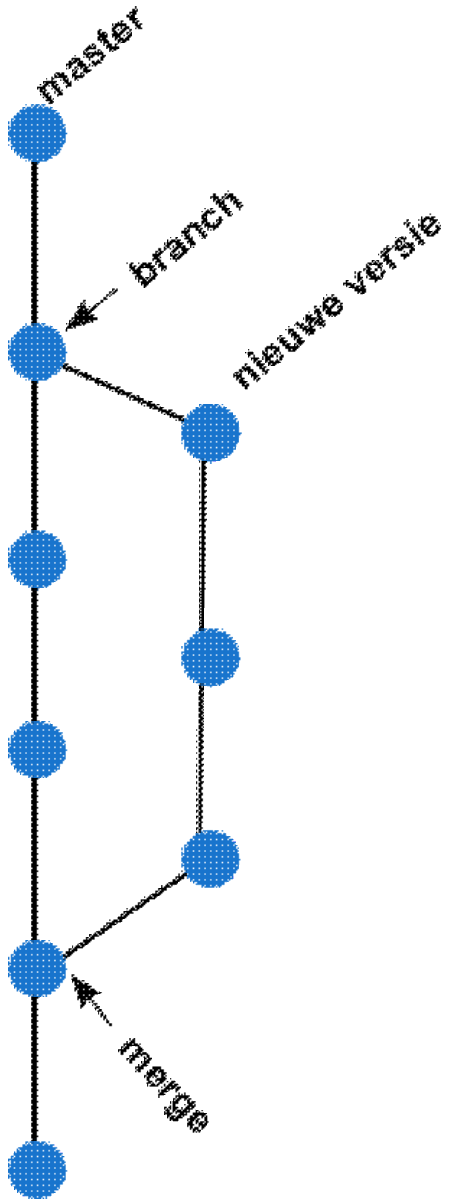
wat ondersteunende scripts:

- vergelijk lokale situatie met de webserver
- signaleer niet-aanwezige of veranderde php-sources via sha1sum

Versiebeheer

behoefte aan versiebeheer
gekozen voor git (www.git-scm.com)

- bijhouden en kunnen terugdraaien van wijzigingen
- branches tbv nieuwe functionaliteit/uitprobeersels
  'productie'-source blijft beschikbaar
- wijzigingen in een branch mergen met de productie-source
- en alles kan op de commandline (maar het hoeft niet)



Vragen?