

Intro

- Siden sidst: evaluering på opgaver og virtuel kursus

Databasestøttet webpublicering

Forelæsning 3

- Hvad har vi lært og hvad kan vi bruge det til?
- Funktioner – dem vi selv laver og alle de som allerede findes i PHP-
fortolkeren
- Inkluderede filer
- Debugging og fejlhåndtering

Hvad har vi lært?

- Client/server-konceptet – f.eks webserver m. php-fortolker som server til en webbrowser over internettet.
- PHP-scripting-begreber og -syntaks
de grundlæggende teknikker i PHP-syntaksen
- HTML: HTML-formularers anvendelse som brugerflade ved input til PHP.

Hvad kan vi bruge det til?

Æblegrødsopgave f.eks – men med små modifikationer meget rige webbaserede services

- Modtage data f.eks. i form af variabler eller arrays fra interface.

Funktioner

- Hvorfor: for at kunne fremstille rutiner (*procedurer*) der kan gentages mange gange – og dermed undgå at skulle skrive en masse kode.
- Anvendelse af *function* er en måde hvorpå man kan opbygge sit script i moduler.

PHP-funktioner

- Der er udviklet utallige
- Se **PHP Manual!**
- Forskellen på egne funktioner og PHP-funktioner?
PHP-funktioner er "black boxes" - du kan ikke se *hvordan* de udfører deres opgave.

Inkluderede filer

- Hvorfor: IGEN - for at kunne genbruge kode der løser hyppige opgaver.

```
include ("nogetsomjeggernevilhavemed.txt")  
include("functions.php")
```

```
include("common.inc")
```

Gode arbejdsforhold

- God arbejdsstation
- Gode rutiner:
 - godt værktøj
 - genvejstaster
 - gode fingre
- God kommunikation
 - med organisationen
 - med maskinen (dvs gode brugerflader til: webserver, PHP-
fortolker, browser m.m.)

Debugging

For at forkorte produktionstiden!!!

- 1) Udskriv strenge: `echo "HALLO - 1"`
- 2) Udskriv variabler: `echo '$ord = ' . $ord . '
';`
- 3) Sæt variabler! : i stedet for

```
$Value = getValue();
```

så overskriv ved at *hardcode* en værdi nedenunder;

```
$strValue = getValue();
```

```
$strValue = "HALLO"
```

- 4) Skriv gode kommentarer i toppen af hver fil og hver funktion.

Fejlhåndtering

- Hvad nu hvis applikationen anvendes forkert?

Vær forberedt på modtagelse af forkerte input!

Regulære udtryk

- **Regulære udtryk** er en syntaks for at genkende *mønstre* i datainput.
- **To eksempler:**
 - Check af email
 - Check af datoformat