Back End-utveckling i PHP

50 YHP

FORTSÄTTNING PHP

Torsdag 1 Februari 2018

- **Förmiddagen:**
  Teori och demonstrationer.

- **Eftermiddagen:**
  en övning på hela teorin, avslutas med genomgång.

- ...handledning som vanligt.

what we know so far...

- **Input** -> PHP Parsning -> **Output** -> Annan parser
- Öppnings- och stängningstaggarna **<?php ?>**
- "värden" och $variabler //även kommentarer
- operationer **+ - = .= <>**
- Några enkla statements (for, echo, switch, while)

what we know so far...

- Hur man gör funktioner
- Hur man jobbar med arrays i PHP

PHPs utvecklingsserver

- PHP installationen innehåller en server för utveckling
- Startas med > *php -S hostname:port*
- e.g. **php -S localhost:8080**
- Startas i en mapp och skickar alla filer som ligger i mappen till webbläsaren, alla .php filer går via PHP parsern

varför ser jag (inte) PHP varningar?

- PHP använder en **php.ini** fil för att hålla reda på alla
- Få ut alla nuvarande inställningar:
  Anropa funktionen phpinfo(); i en PHP fil.
- Vart finns min **php.ini** fil?
  Kör > php --ini i en terminal.

varför ser jag (inte) PHP varningar?

- ini filen kan innehålla **display_errors = on** eller **display_errors = off**
- Ini filen kan läsas och ändras efter behov i kod med: `ini_set()` och `ini_get()`

```
// men använd 1, 0 inte "on", "off"
```

- *E.g. för att se v*ilken felrappoteringsnivå som är satt:

```
echo ini_get("error_reporting");
```

Funktioner

- En funktion skapas med keywordet **function**
- Koden skriven i funktionen körs först när man *anropar* eller *kallar* på funktionen
- Koden som skrivs innanför funktionens {} blir ett eget *local scope* **och har ingen kännedom om globalscope**
- Värden kan skickas in i en funktion som argument **(inställningar)**
- Ett värde kan skickas ut ur en funktion med keywordet *return* **(ett anrop översätts till return värdet)**

Funktioner

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

Ger variabeln firstName i globalscope värdet "lisa"

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

Vi ska ge variabeln myName i globalscope ett värde...
vi anropar fullName funktionen för att ta reda på vilket värde anropet retunerar

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

Vi kör nu fullName funktionen, argumentet $firstName i localscope är inställd på värdet av $firstName i globalscope

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

"Lisa"

$firstName (globalscope)

"simpson"

```
1. $firstName
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

Variabeln fullName i localscope får värdet av firstName ihopsatt med familyName som båda finns i localscope

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

"Lisa"

"simpson"

"Lisa simpson"

Funktioner

vi ser keywordet return och skickar tillbaka värdet av den lokala variabeln fullName

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

"Lisa simpson"

Funktioner

Vi återvänder till globalscope och funktionens localscope tas bort
return värdet är det enda som överlever

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = fullName($firstName, "simpson");
```

Funktioner

Variabeln $myName sätts till värdet Lisa simpson

```
1. $firstName = "Lisa"
2.
3. function fullName($firstName, $familyName) {
4.   $fullName = $firstName." ".$familyName;
5.
6.   return $fullName;
7. };
8.
9. $myName = "Lisa simpson"
```

Scope

Varje PHP fil har ett **global scope**.

Varje funktion har ett **local scope**. Detta scope återställs för varje gång funktionen kallas på (varje anrop har ett eget scope)

Inga variabler existerar mellan dessa scope automatiskt d.v.s. globala variabler finns inte inuti era funktioner!

Övning: Global Scope

- Skapa en ny **php fil** med två styckten php styckten (<?php ?>)
- I det första stycket skapar ni en variabel och ger den värdet "foobar"
- Finns denna variabel i det andra stycket? Har den värdet foobar eller något annat?
- **Skrik inte ut svaret!** 🤫

Arrays

I PHP skapas listor / arrayer med en funktion som heter **array**

```
$myArray = array("first", "second", "third");
```

värden läses ut som i JS

```
$myArray[0] //"first"
```

I PHP får man ut hur lång en array är genom att skicka den till funktionen **count**

```
$length = count($myArray);
```

Arrays

I PHP skapas listor / arrayer med en funktion som heter **array**

```
$myArray = array("first", "second", "third");
```

värden läses ut som i JS

```
$myArray[0] //"first"
```

I PHP får man ut hur lång en array är genom att skicka den till funktionen **count**

```
$length = count($myArray);
```

Arrays

"Strängar" i PHP är också Arrays

count() retunerar längden för en sträng och
för att hämta ut en symbol kan [] användas

```
"myTextString"[2]; // T
"myTextString"[6]; // S

count("myTextString"); // 12
```

Arrays

I PHP kan man enkelt sortera sina arrays med följande funktioner:

- `$array = sort($array)`
  `// sorterar lågt till högt`

- `$array = rsort($array)`
  `// sorterar i omvänd,"reversed", ordning`

konstanter

Konstanter är värden som kan vara dynamiska men som inte kan ändras

Bra exempel är inställningar, produktnamn, versionsnummer...

variabel = i varians, i förändring.

konstant = konstant likadana.

konstanter

Initieras i PHP med funktionen **define**:

```
// define(namn, värde)

define("IS_TESTING", true);

if (IS_TESTING) {

 echo "This is a test";

}
```

print och print_r

- **print** fungerar exakt likadant som **echo**
- både **print** och **echo** finns både som keywords och som funktioner
- **print** och **echo** kan bara skriva strängar till output
- **print_r()** är en funktion som kan skriva all typ av data på ett sätt som vi kan läsa det - bra främst vid utveckling och testning.

importera filer

I php kan man precis som i NodeJS importera andra filer och moduler

Det finns två stycken keywords för detta:

- Include
- Require

- Dom används likadant:

```
include "sökväg/till/filen.php";

require "sökväg/till/filen.php";
```

importera filer

Om **require** används istället för **include**, så kraschar programmet om filen inte finns.

Mitt tips: använd require om ni inte har en anledning att göra annorlunda, då är det enklare att märka fel.

importera filer

När en fil importeras läggs **hela den filens innehåll** på den raden include/require är skriven!

- **Variabler hamnar i filens globalscope!**
- **Funktioner blir tillgängliga.**
- *Parsen kommer fortsätta på rad 1 i den importerade filen.*
- *Förutom .php filer kan importera exempelvis .html och .js filer också (allt går).*
  bara PHP filer parsas, textfiler skickas som echo

importera filer

Sökvägar i PHP kan börja på / för att utgå från serverns "root" mapp

- Kan vara en särskild mapp, ibland kallad wwwroot
- I utvecklingsserverns fall: samma som hårddisken. */ på mac*, *C:\ på windows*

importera filer

Sökvägar i PHP kan börja på ./ för att utgå från samma mapp som .php filen ligger i, eller ../ för att börja från den överliggande mappen.

Man kan även lägga flera ../ på raden:

../../../

skulle betyda tre mappar uppifrån denna 📦

Dagens övning

- Du ska göra en liten **ranking-website**
  kan vara för musik, *böcker*, *film eller annat*.

- ***Projektet*** ska ha tre sidor:
  start - med en liten introduktion
  listan - som innehåller själva listan
  kontakt - med presentation och namn på dig som gjort sidan

- Alla sidorna ska använda samma header och footer som ska importeras in

- Listan ska vara en array som innehåller strängar som ser ut så här: "12. Titanic" eller "9. Jurassic Park" - alltså som börjar på en siffra, siffran är en **poäng**

- En funktion ska finnas som sorterar och skriver ut listan med den som har **högst poäng** överst

- funktionen ska finnas i en egen php fil, inte samma som listan (importera den filen sen för att skriva ut)

- Sidan ska ha ett speciellt namn, exempel: Gurkor.nu, Använd en konstant för att bestämma namnet på sidan.
  Namnet ska visas minst tre gånger.

- **När ni är klara ska så ladda upp sidan på er FTP server och skicka länken till mig på slack (gärna som PM)**

- **Ingen stress** 🙏

Tack för idag!