

Introduktion till PHP

... och byggandet av en simpel gästbok



PHP

Learn to program in PHP, a widespread language that powers sites like Facebook.

START

900k+
enrolled students

4 Hours
estimated course time

Beginner
required technical level

WELCOME TO PHP!

0%

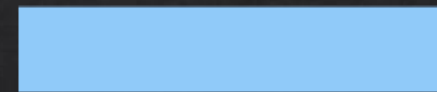
Introduction to PHP

This tutorial will introduce you to PHP, a server-side scripting language you can use to make dynamic websites and web applications.



Dagens agenda – pass 1

- ◊ Introduktion till PHP
 - ◊ Hur används vi PHP?
 - ◊ Variabler
 - ◊ Loopar & iterationer
 - ◊ Funktioner
 - ◊ Namespace
 - ◊ Klasser
 - ◊ Exempel!
- ◊ Kursens github-repo
- ◊ Vi bygger en gästbok
- ◊ Del 2 – Johan: Composer och DM



Användning av PHP (1)

```
1  <?php
2
3  echo "This is an awesome script!";
4
```

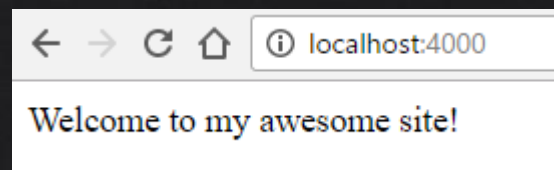


```
php index.php
This is an awesome script!
```

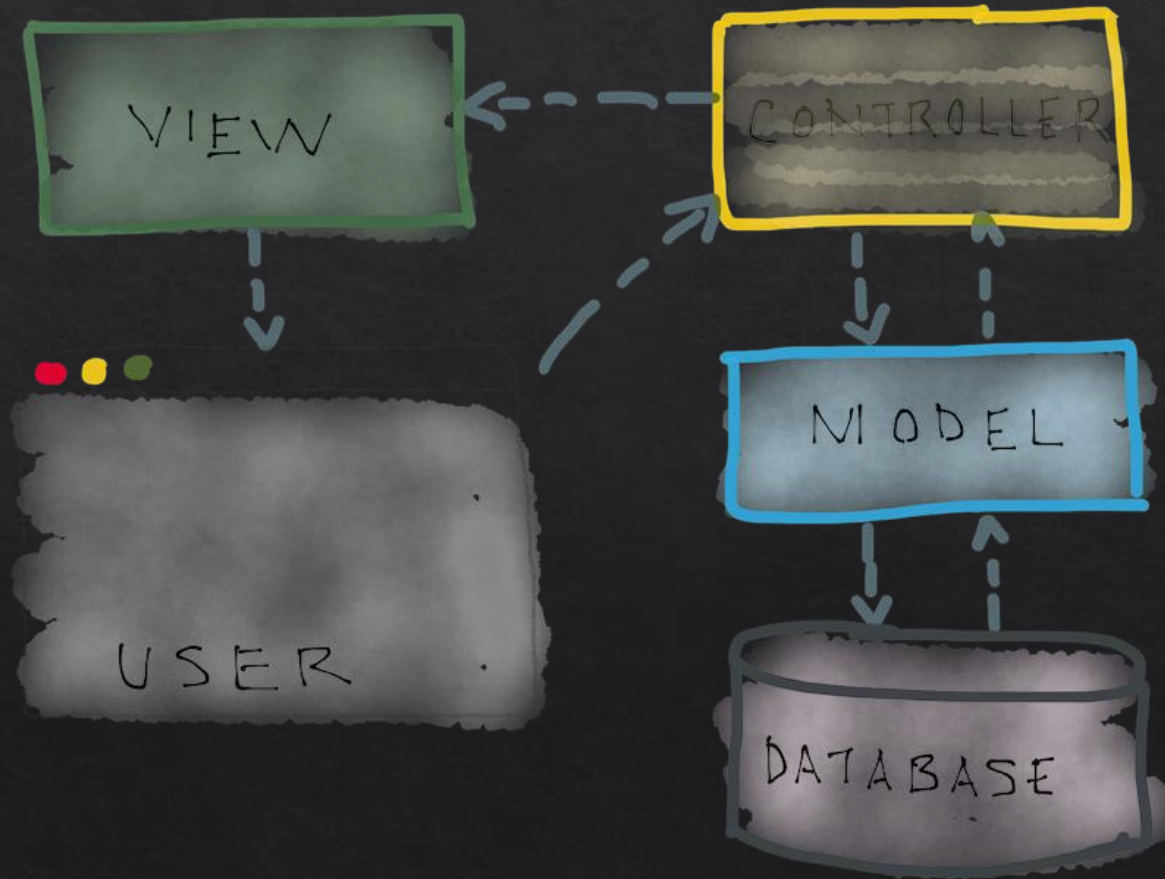

Användning av PHP (2)

```
1  <!doctype html>
2  <html>
3  <head>
4      <title>Example document</title>
5  </head>
6  <body>
7      <?php
8          echo "Welcome to my awesome site!";
9      ?>
10 </body>
11 </html>
```

```
<!doctype html>
<html>
<head>
    <title>Example document</title>
</head>
<body>
    Welcome to my awesome site!
</body>
</html>
```



Användning av PHP (3)



Användning av PHP (3 forts.)

```
/*
 * Return index page
 */
$app->get ( '/', function () use($app, $db) {
    $app->response->setStatus ( 200 );
    $app->response->headers->set ( 'Content-Type', 'text/html' );
    $movies = $db->select ( "movie", "*", [
        "ORDER" => 'id DESC'
    ] );
    $app->render ( 'index.php', array (
        "movie" => $movies [0],
        "movies" => $movies
    ) );
} );
```



```
<h1>Movies</h1>
<hr>
<table class="table table-striped">
    <tr>
        <th>Id</th>
        <th>Title</th>
        <th>Genre</th>
        <th>Length</th>
        <th>IMDB</th>
    </tr>
    <?php
    foreach($movies as $movie){
        echo "<tr>";
        echo "<td>".$movie['id']."</td>";
        echo "<td>".$movie['title']."</td>";
        echo "<td>".$movie['genre']."</td>";
        echo "<td>".$movie['length']."</td>";
        echo "<td><a href='". $movie['imdb']."'>IMDB</a></td>";
        echo "</tr>";
    }
    ?>
</table>
```


Över till hur man skriver PHP!

Variabler och utskrifter

- ◊ Variabler behöver inte deklareras innan de tilldelas värde
- ◊ PHP är ett otypat språk – datatyp behöver inte anges
- ◊ Variabelnamn ska börja \$, följt av en bokstav, eller "_" (underscore)
- ◊ Variabelnamn får bara innehålla bokstäver, siffror eller "_" (underscore)
- ◊ Man använder "camelCase" vid namngivning av variabler

```
1  <?php
2
3  $name = "Anton";
4  $age = 27;
5
6  echo $name." is ".$age." years old";
7  // or
8  echo "$name is $age years old";
9
```

Kommentarer

```
1  <?php
2
3  // One line comment
4
5  /*
6     Multiple line comment
7  */
8
```

if, elseif, else

```
1  <?php
2
3  if ($expr1) {
4      // $expr1 is true
5  } elseif ($expr2) {
6      // $expr2 is true
7  } else {
8      // else
9  }
10
```

switch

```
1  <?php
2
3  switch ($expr) {
4      case 0:
5          echo 'First case';
6      case 1:
7          echo 'Second case';
8      case 3:
9          echo 'Third case';
10     default:
11         echo 'Default case';
12 }
13
```


for-loop

```
1  <?php
2
3  for ($i = 0; $i < 10; $i++) {
4      // for body
5  }
6
```

foreach

```
1  <?php
2
3  foreach ($iterable as $key => $value) {
4      // foreach body
5  }
6
7
```

while

```
1  <?php
2
3  while ($expr) {
4      // structure body
5  }
6
7
```

do while

```
1  <?php
2
3  do {
4      // structure body;
5  } while ($expr);
6
7
```

try catch

```
1  <?php
2
3  try {
4      // try body
5  } catch (FirstExceptionType $e) {
6      // catch body
7  } catch (OtherExceptionType $e) {
8      // catch body
9  }
10
11
```


Arrayer

```
1  <?php
2
3  $numbers = [1, 2, 3, 4, 5];
4
5  foreach ($numbers as $number) {
6      echo $number . "<br>";
7  }
8
```

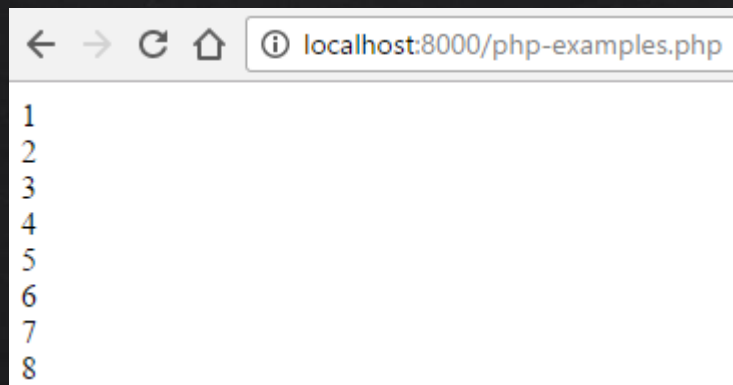


← → ↻ 🏠 ⓘ localhost:8000/php-examples.php

```
1
2
3
4
5
```

Arrayer

```
1  <?php
2
3  $numbers = [1, 2, 3, 4, 5];
4  $numbers[] = 6;
5  $numbers[] = 7;
6  $numbers[] = 8;
7
8  foreach ($numbers as $number) {
9      echo $number . "<br>";
10 }
11
```



Arrays

```
1 <?php
2
3 $numbers = [1, 2, 3, 4, 5];
4 $numbers[] = 6;
5 $numbers[] = 7;
6 $numbers[] = 8;
7
8 unset($numbers[5]); // Key 5 => number 6
9
10 foreach ($numbers as $number) {
11     echo $number . "<br>";
12 }
13
```

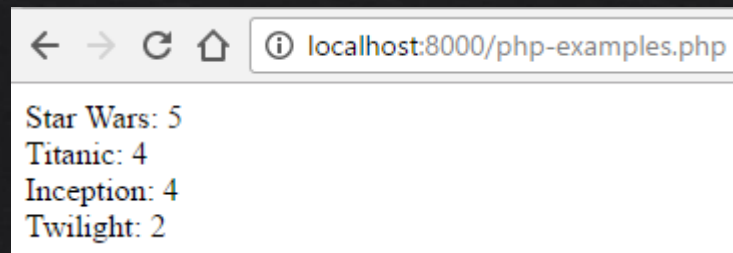


← → ↻ 🏠 ⓘ localhost:8000/php-examples.php

```
1
2
3
4
5
6
7
8
```

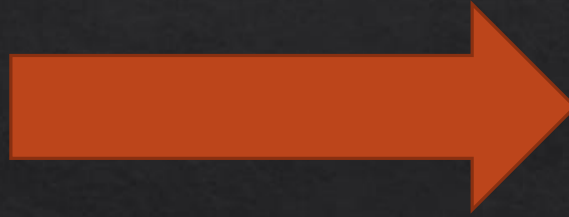
Arrayer med nycklar

```
1  <?php
2
3  $movies = [
4      "Star Wars" => 5,
5      "Titanic" => 4,
6      "Inception" => 4,
7  ];
8
9  $movies["Twilight"] = 2;
10
11 foreach ($movies as $movie => $grade) {
12     echo "$movie: $grade <br>";
13 }
14
```



Klasser

```
1  <?php
2
3  class Movie {
4
5      private $title;
6      private $year;
7      private $grade;
8      private $actors;
9
10     public function __construct($title, $year)
11     {
12         $this->title = $title;
13         $this->year = $year;
14         $this->grade = null;
15         $this->actors = [];
16     }
17
18     public function getJSON()
19     {
20         return json_encode([
21             "title" => $this->title,
22             "year" => $this->year,
23             "grade" => $this->grade,
24             "actors" => $this->actors
25         ]);
26     }
27 }
28
29 $movie = new Movie("Star Wars", 1977);
30 echo $movie->getJSON();
31
```



← → ↻ 🏠 ⓘ localhost:8000/php-examples.php



{"title":"Star Wars","year":1977,"grade":null,"actors":[]}

Namespace

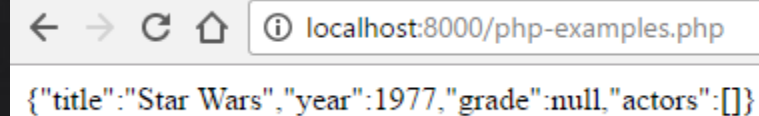

- ◊ Vid mycket kod vill vi se till att vår kod inte kolliderar!
 - ◊ Vad händer om vi har två klasser som heter samma sak?
- ◊ Vi vill kunna bygga fristående komponenter, som fungerar med andra komponenter!
 - ◊ Så att vi kan återanvända (och använda andras) komponenter i olika projekt.



```
movie.php
1  <?php
2
3  namespace App;
4
5  class Movie {
6
7      private $title;
8      private $year;
9      private $grade;
10     private $actors;
11
12     public function __construct($title, $year)
13     {
14         $this->title = $title;
15         $this->year = $year;
16         $this->grade = null;
17         $this->actors = [];
18     }
19
20     public function getJSON()
21     {
22         return json_encode([
23             "title" => $this->title,
24             "year" => $this->year,
25             "grade" => $this->grade,
26             "actors" => $this->actors
27         ]);
28     }
29 }
```



```
$movie = new App\Movie("Star Wars", 1977);
echo $movie->getJSON();
```



```
localhost:8000/php-examples.php
{"title":"Star Wars","year":1977,"grade":null,"actors":[]}
```

```
movie.php    index.php
1  <?php
2
3  namespace App;
4
5  class Movie {
6
7      private $title;
8      private $year;
9      private $grade;
10     private $actors;
11
12     public function __construct($title, $year)
13     {
14         $this->title = $title;
15         $this->year = $year;
16         $this->grade = null;
17         $this->actors = [];
18     }
19
20     public function getJSON()
21     {
22         return json_encode([
23             "title" => $this->title,
24             "year" => $this->year,
25             "grade" => $this->grade,
26             "actors" => $this->actors
27         ]);
28     }
29 }
```

```
movie.php    index.php
1  <?php
2
3  require("movie.php"); // <= Jobbigt!
4
5  use App\Movie;
6
7  $movie = new Movie("Star Wars", 1977);
8  echo $movie->getJSON();
9
10
```




```
← → ↻ 🏠 ⓘ localhost:8000/php-examples.php
{"title":"Star Wars","year":1977,"grade":null,"actors":[]}
```

movie.php

index.php

```
1 <?php
2
3 require("movie.php"); // <= Jobbigt!
4
5 use App\Movie;
6
7 $movie = new Movie("Star Wars", 1977);
8 echo $movie->getJSON();
9
10
```

```
1 <?php
2
3 require("movie.php"); // <= Jobbigt!
4
5 use App\Movie as M;
6
7 $movie = new M("Star Wars", 1977);
8 echo $movie->getJSON();
9
```



PHP – Style guidelines

<http://www.php-fig.org/>

Hantera formulärsdata

Hantera data från GET-anrop

- ◇ Men HTTP-anrop (GET) kan man skickar med parametrar, t.ex.
 - ◇ index.php?course=DA287A



```
movie.php | index.php
1 <?php
2
3 $course = $_GET['course'];
4 echo $course; // Prints "DA287A"
5
6
```

Hantera data från GET-anrop

- ◇ Men HTTP-anrop (GET) kan man skickar med parametrar, t.ex.
 - ◇ `index.php?course=DA287A&courseResponsible=Anton`



```
movie.php | index.php
1  <?php
2
3  $course = $_GET['course'];
4  $courseResponsible = $_GET['courseResponsible'];
5  echo $course; // Prints "DA287A"
6  echo $courseResponsible; // Prints "Anton"
7
8
```

Hantera data från POST-anrop

- ❖ Till skillnad från GET (som används för att efterfråga data) så används POST för att skicka med data. Detta görs t.ex. genom ett HTML-formulär:

```
<form action="index.php" method="post">
  <label for="course">Course</label>
  <input type="text" name="course" id="course">
  <label for="courseResponsible">Course responsible</label>
  <input type="text" name="courseResponsible" id="courseResponsible">
</form>
```

Course Course responsible

- ❖ Och tas sedan emot genom \$_POST-funktionen i PHP.

```
1 <?php
2
3 $course = $_POST['course'];
4 $courseResponsible = $_POST['courseResponsible'];
5 echo $course; // Prints "DA287A"
6 echo $courseResponsible; // Prints "Anton"
7
8
```

Vi bygger en gästbok!

Kursmoment

1. PHP & Composer



2. MVC & Lumen/Laravel



3. Testning & deployment

