PHP-kielen perusteiden tiivistelmä

Tämä luku sisältää PHP-kielen perusteet erittäin tiiviisti ilmaistuna. Oletuksena on, että lukijalla on esitietoina tavanomaisen ohjelmoinnin perusteet -opintojakson sisältö.

PHP-koodin merkintä ja kommentit

PHP-ohjelmakoodi merkitään (upotetaan) HTML-koodin sekaan tavanomaisesti seuraavalla XML-prosessointiohjeella

```
<h1>HTML-merkkausta</h1>
<?php
echo "XML-prosessointiohje on suositeltavin";
?>
<p>HTML-merkkaus jatkuu</p>
```

Kommentit merkitään seuraavasti

```
<?php
echo "Testiteksti"; // Rivin loppuosa kommenttia

/* Tämä on
   monirivinen
   kommentti */

echo "Testiteksti"; # Rivin loppuosa kommenttia
?>
```

Muuttujat

- Muuttujiin viitataan tunnuksella `$muuttuja` ja dollarimerkki muuttujan nimen alussa on pakollinen

PHP on heikosti (dynaamisesti) tyypitetty kieli eli

- Muuttujia ei tarvise luoda erikseen
- Muuttujan tila varataan, kun sitä käytetään ensimmäisen kerran (esim. sijoituslause)
- Muuttujille ei määritellä tietotyyppiä
- Muuttujaan voi tallentaa mitä tahansa tietotyyppiä olevia arvoja
- Muuttujan tietotyyppi riippuu tallennetusta arvosta ja saman muuttujan tietotyyppi voi vaihdella ohjelman suorituksen aikana.

Esimerkkejä muuttujista

```
1  $muuttujanArvo = "Ari";
2  $muuttujanarvo = "Jari";
3  $muuttujan_arvo = "Kari";
4  $kokolukumuuttuja = 42;
5  $liukulukumuuttuja = 42.18;
6  $totuus = true;
```

Taulukkomuuttujat

- Taulukon alkioon viitataan joko **kokonaislukuindeksillä** tai **merkkijonoavaimella**

```
1  $taulu[0] = "Eka";
2  $taulu[1] = "Toka";
3  $taulu["sukunimi"] = "Rantala";
4  $taulu["etunimi"] = "Ari";
```

Taulukon alustaminen sijoittamalla

```
1  $ruoka[] = "Tomaatti"; // $ruoka[0] = "Tomaatti"
2  $ruoka[] = "Pinaatti"; // $ruoka[1] = "Pinaatti"
3  $ruoka[] = "Peruna";   // $ruoka[2] = "Peruna"
4  $ruoka[] = "Salaatti"; // $ruoka[3] = "Salaatti"
```

Taulukon alustaminen array-käskyllä

```
1  $ruoka = array("Tomaatti", "Pinaatti", "Peruna", "Salaatti");
2  print_r($taulu); // print_r tulostaa taulukon avaimineen
3
4  $menu = array("alkuruoka" => "Salaatti",
5                "paaruoka" => "Pihvi",
```

```
                "jalkiruoka" => "Kiisseli");
print_r($menu); // print_r tulostaa taulukon avaimineen
```

> Käyttäessäsi merkkijonoa avaimena kirjoita aina heitto- tai lainausmerkit näkyviin, jollet tiedä miksi ne voisi jättää pois!

Taulukot voivat olla moneen suuntaan moniulotteisia esim. seuraavat ovat ok

```
$taulu[0]['nimi'] = "Ari";
$taulu[1]['nimi'] = "Sepi";
$taulu['nimi'][2][10] = "Hanski";
```

Etukäteen määritellyt muuttujat

- PHP-skripteillä on käytössä ympäristöstä riippuva määrä erilaisia etukäteen määriteltyjä muuttujia. Ympäristöön vaikuttaa käyttöjärjestelmä, web-palvelin, PHP ja niiden eri asennustavat ja versiot. Näistä muuttujista osan luo web-palvelin ja osan PHP itse. Täyden listan oman ympäristösi etukäteen määritellyistä muuttujista saat tulostettua funktiolla `phpinfo()`. Tässä on lyhyesti listattuna muutamia.
- Nämä etukäteen määriteltyjen muuttujien arvot ovat käytettävissä nk. superglobaaleissa assosiatiivisissa (merkkijonoavaimin viitattavissa) taulukoissa -> Arvot käytettävissä funktioissa ilman eri määrittelyjä.

$_GET

Muuttujat, jotka ovat välitetty skriptille GET-metodilla. Esim. $_GET['sukunimi']

$_POST

Muuttujat, jotka ovat välitetty skriptille POST-metodilla. Esim. $_POST['sukunimi']

$_COOKIE

Muuttujat, jotka ovat välitetty skriptille evästeinä. Esim. $_COOKIE['login']

$_REQUEST

Muuttujat, jotka ovat välitetty skriptille GET- ja POST-metodeilla tai evästeinä.

$_SESSION

Muuttujat, jotka ovat välitetty skriptille käyttäen PHP:n sisäänrakennettua istunnonhallintamekanismia. Esim. $_SESSION['login']

$_FILES

Muuttujat, jotka ovat välitetty skriptille File Upload-menetelmällä. Esim. $_FILES['filetto']

$_SERVER

web-palvelimen generoimia muuttujia

$_ENV

Muuttujat, jotka skripti on perinyt ajoympäristöstänsä.

Ajamalla seuraavan ohjelman saat hyvän käsityksen siitä mitä kaikkea em. muuttujilla on tarjota

```
<title>PHP-esimerkki phpinfo.php</title>
<?php
phpinfo();
?>
```

Tietotyypit

Totuusarvo

```
$mja = False; // Sijoitetaan totuusarvo FALSE muuttujalle $mj
$mja = TRUE; // Merkkikoolla ei ole väliä,
```

Kun jonkin muun tyypin muuttujaa muutetaan totuusarvoksi, niin seuraavat arvot muunnetaan boolean-tyypin FALSE-arvoksi:

- boolean: FALSE
- integer: 0
- float: 0.0
- string: tyhjä merkkijono ja merkkijono "0"
- array: tyhjä taulukko, ei alkioita
- object: olio ilman asetettuja ominaisuuksia
- NULL

Kaikki muut arvot muutetaan TRUE-arvoksi!

Kokonais- ja liukuluvut

```php
$mja = -12; // -12 kymmenjärjestelmässä
$mja = 012; // Oktaaliku 12 (kymmenjärjestelmässä 10)
$mja = 0x12; // heksadesimaaliluku 12 (kymmenjärjestelmässä 18
$mja = 1.2;
$mja = 1.2e3; // 1200
$mja = 3E-5; //  0.00003
```

Merkkijonot

Kolme tapaa määritellä

```php
$str1 = 'Tämä on heittomerkein määritelty merkkijono';
$str2 = "Tämä on lainausmerkein määritelty  merkkijono";

$str3 = <<<EOD
Tässä on
kolmas kolmirivinen
merkkijono heredoc-syntaksilla.
EOD;
```

Esimerkit

```php
// Heittomerkit
$mja = 36;
$str = 'James O'Neill is $mja years old';
echo $str; //Tulostaa: James O'Neill is $mja years old

// Lainausmerkit
$mja = 54;
$_POST['nimi'] = "Ari";

// Seuraavat rivit tulostavat kumpikin
// Ari on 54-vuotias
echo $_POST['nimi'] . " on " . $mja . "-vuotias";
echo "{$_POST['nimi']}  on $mja-vuotias";

// Heredoc-syntaksi
$mja = 54;
$_POST['nimi'] = "Ari";

```

```php
$lomake = <<<EOLomake
<form action="softa.php" method="get">
<input type='text' name="nimi" value="{$_POST['nimi']}">
<input type='text' name="ika" value="$mja">
<input type='submit' name="nappi">
</form>
EOLomake;

// Tulostaa lomakkeen muuttujien arvoineen
echo $lomake;
```

Operaattorit

Aritmeettiset operaattorit

```php
$a+$b;
$a-$b;
$a*$b;
$a/$b;
$a%$b;

// Tulostaa: 4
$a=3; echo ++$a;

// Tulostaa: 3
$a=3; echo $a++;

// Tulostaa: 2
$a=3; echo --$a;

//Tulostaa: 3
$a=3; echo $a--;
```

Sijoitusoperaattorit

```php
$a = 36;
$a += 4; //kuten $a=$a+4;
$a -= 4; //kuten $a=$a-4;
$a *= 4; //kuten $a=$a*4;
$a /= 4; //kuten $a=$a/4;
```

```
$a %= 4; //kuten $a=$a%4;

$s1 = "Aku";
$s1 .= " Ankka";
//Tulostaa: Aku Ankka
echo $1;
```

Vertailuoperaattorit

```
("36" == 36) // TRUE
("36" === 36) // FALSE
(12 < 36) // TRUE
(12 > 36) // FALSE
(36 <= 36) // TRUE
(36 >= 36) // TRUE
(12 <> 36) // TRUE
(12 != 36) // TRUE
(12 !== 36) // TRUE
("36" !== 36) // TRUE
```

Loogiset operaattorit

```
AND ja &&
OR ja ||
XOR
!
```

Muut operaattorit

```
$str = "No" . " hei!";
echo $str; // "No hei!"

print(36/0); //Tulostaa virheen
@print(36/0); //Ei tulosta mitään

$str = `netstat -an`;
echo "<pre>$str</pre>";
```

Aliohjelmat eli funktiot

Pari esimerkkiä

alv-esimerkki.php

```
<title>ALV-esimerkki</title>
<?php
// Pääohjelma, ohjelman suoritus alkaa tästä
$alviton_loppusumma = 100;

// Funktion anna_alv kutsu:
$arvonlisavero = anna_alv($alviton_loppusumma);

// Lisätään ja tulostetaan arvonlisäverollinen hinta
$alvillinen_hinta = $alviton_loppusumma + $arvonlisavero;
echo "ALVillinen hinta on: $alvillinen_hinta";    // Tulostaa

// Funktion anna_alv määrittely
function anna_alv($maara) {
   $alv = $maara * 0.24;
   return $alv;
}
?>
```

muuta-saldoa.php

```
<title>Parametrin välittäminen viittauksena -esimerkki</title
<?php
// muuta-saldoa.php

// Pääohjelma
$saldo  = 100;
$muutos = -40;

// $saldo välitetään nyt viittauksena, koska
// niin on määritelty funktion esittelyssä
// Huomaa, että funktion palauttamaa NULL-arvoa
// ei käytetä (sijoiteta) mihinkään.

muuta_saldo($saldo, $muutos);
echo "Saldo: $saldo<br>
";  // tulostaa: 60

// Vähennetään $saldoa toistamiseen
```

```php
muuta_saldo($saldo, $muutos);
echo "Saldo: $saldo<br>
";    // tulostaa: 20

// Huomaa &-merkki muodollisen parametrin edessä: Viittaus
function muuta_saldo(&$local_saldo, $muutos) {
   $local_saldo = $local_saldo + $muutos;
   // Täällä ei ole return-lausetta, palautetaan siis NULL
}

?>
```

Staattinen muuttuja

static-counter.php

```php
<title>Staattinen muuttuja laskurina</title>
<?php
// static-counter.php

function huonoLaskuri() {
   if (isset($count)) {
      $count++;
   } else {
      $count = 0;
   }
   echo "huonoLaskuri: $count<br>";
}

function hyvaLaskuri() {
   static $count = 0; // sijoitetaan 0 ensimmäisellä kerralla
   $count++;
   echo "hyvaLaskuri: $count<br>";
}

// Molempia funktioita kutsutaan erikseen 3 kertaa
for ($i = 1; $i <= 3; $i++) {
   huonoLaskuri();
}

for ($i = 1; $i <= 3; $i++) {
```

```
    hyvaLaskuri();
}
?>
```

Ohjelman tulostus on seuraava:

huonoLaskuri: 0
huonoLaskuri: 0
huonoLaskuri: 0
hyvaLaskuri: 1
hyvaLaskuri: 2
hyvaLaskuri: 3

Ohjausrakenteita

IF-esimerkki

```
<title>Esimerkki If-rakenteesta</title>
<?php
// if-esimerkki.php
$nimi = "Kaveri";
$ika = 36;
if ($ika < 0) // Jos ika on pienempi kuin nolla {
   echo "$nimi ei vielä ole syntynyt
";
}
else if ($ika < 18) { // muutoin jos ika on pienempi kuin 18
   echo "$nimi ei ole täysi-ikäinen
";
}
else {
   echo "$nimi on täysi-ikäinen
";
}
?>
```

Switch-esimerkki

```
<title>Esimerkki Switch-rakenteesta</title>
<pre>
<?php
// switch-esim.php
$n1 = 36;
$n2 = 10;
$action = "erotus";

switch ($action)
{
   case "summa": // Jos muuttujan $action arvo on "summa", su
      echo $n1, " + ", $n2, " = ", $n1 + $n2, "
";
      break;
   case "erotus":
      echo $n1, " - ", $n2, " = ", $n1 - $n2, "
";
      break;
   case "tulo":
      echo $n1, " * ", $n2, " = ", $n1 * $n2, "
";
      break;
   case "osamaara":
      echo $n1, " / ", $n2, " = ", $n1 / $n2, "
";
      break;
   default:
      echo "Virhe: Laskutoimitusta ei ole määrittelty";
      break;
}
?>
</pre>
```

Ohjelma tulostaa:
36 - 10 = 26

For-esimerkki

```
<title>Taulukon läpikäynti For-rakenteella</title>
<?php
```

```php
//for-esimerkki.php

$ruoka[] = "Tomaatti"; //ruoka[0]
$ruoka[] = "Pinaatti"; //ruoka[1]
$ruoka[] = "Salaatti"; //ruoka[2]
$ruoka_lkm = count($ruoka);

for($i = 0; $i < $ruoka_lkm; $i++) {
   print("Ruoka $i on
   $ruoka[$i]<br>
");
}
?>
```

Foreach-esimerkit

```php
<title>Taulukon läpikäynti Foreach-rakenteen avulla</title>
<?php
//foreach-esim1.php

$taulu[0] = "Rantala";
$taulu[1] = "Rasku";
$taulu[2] = "Rautanen";

// Ei tarvitse välittää lukumäärästä:
foreach ($taulu as $arvo) {
    echo "$arvo<br>";
}

?>
```

```php
<title>Hajautustaulukon läpikäynti Foreach-rakenteen avulla</
<?php
// foreach-esim2.php

$taulu["sukunimi1"] = "Rantala";
$taulu["sukunimi2"] = "Rasku";
$taulu["sukunimi3"] = "Rautanen";

```

```php
foreach ($taulu as $arvo) {
    echo "$arvo<br>";
}
?>
```

```php
<title>2-ulotteisen taulukon läpikäynti Foreach-rakenteella</
<pre>
<?php

// foreach-esim3.php

// Tulostetaan 2-ulotteisen taulukon kaikki arvot
// foreach-rakenteella

$taulu[0]["nimi"]        = "Rantala";
$taulu[0]["gsmnro"]      = "040 001";
$taulu[0]["paikkakunta"] = "Jyväskylä";

$taulu[1]["nimi"]     = "Rasku";
$taulu[1]["gsmnro"]   = "040 002";

foreach ($taulu as $uloinavain => $sistaulu) {
   echo "**** Henkilön nro $uloinavain tiedot: ****";
   // Käydään läpi sisemmät taulut vuorollaan:
   foreach ($sistaulu as $avain => $arvo) {
      echo "  $avain = $arvo";
   }
}

?>
</pre>
```

- foreach siirtää aluksi taulukon sisäisen osoittimen automaattisesti alkuun (ei tarvita reset-funktiota)
- foreach käyttää alkuperäisen taulukon kopiota. Tällöin taulukon läpikäynti ei muuta alkuperäisen taulukon sisäisen osoittimen paikkaa

While-esimerkki

```php
<?php
$lkm = 1;
$kerrat = 10;

while ($lkm <= $kerrat) {
   echo "Terve $lkm kerran<br>
";
   $lkm++;
}

?>
```

Do-While -esimerkki

```php
<?php
$lkm = 1;
$kerrat = 10;
do {
   echo "Terve $lkm kerran<br>
";
   $lkm++;
} while ($lkm <= $kerrat);
?>
```

Break -lause

- Break-lause lopettaa toistorakenteen toiston
- Esim. seuraava ohjelma tulostaa "vain" neljä kertaa "Terve...":

```php
<?php
$lkm = 1;
$kerrat = 10;
do
{
   if ($lkm == 5)
   {
      break;
   }
   echo "Terve $lkm kerran<br>
```

```
";
   $lkm++;
} while ($lkm <= $kerrat);
?>
```

> Jätetty tarkoituksella tyhjäksi

```
© #AriRantala
```