Functies

Huub de Beer

Eindhoven, 4 juni 2011

Functies: je kent ze al

Je hebt al verschillende PHP functies gebruikt:

- **pi**() om het getal π uit te rekenen.
- **sin**(0.453) om het de sinus van het getal 0.453 uit te rekenen.
- **count**($punten) om het aantal punten in het array $punten te bepalen..
- dump_var($var) om de waarde en het type van variabele $var af te drukken.

- PHP kent honderden functies
- Je kunt ze vinden in de handleiding
- Er bestaat waarschijnlijk een functie voor

Functies: je kent ze al

Je hebt al verschillende PHP functies gebruikt:

- **pi**() om het getal π uit te rekenen.
- **sin**(0.453) om het de sinus van het getal 0.453 uit te rekenen.
- **count**($punten) om het aantal punten in het array $punten te bepalen..
- dump_var($var) om de waarde en het type van variabele $var af te drukken.

- PHP kent honderden functies
- Je kunt ze vinden in de handleiding
- Er bestaat waarschijnlijk een functie voor

Sommige functies retourneren een waarde

- Invoer: er gaan nul of meer waarden in, dat kunnen ook variabelen zijn
- Verwerking: de functie verwerkt die invoer
- Returnwaarde: de functie “berekent” een returnwaarde en stuurt die waarde terug naar de aanroep.
- Deze functies zijn *altijd* onderdeel van een expressie:
  - Ze staan aan de *rechterkant* van het =-teken
  - Ze maken deel uit van een conditie

```
1 if  (is_numeric( $_POST['r'] )) {
2   $r  = $_POST['r'];
3   $opp = 2 * pi()  * $r;
4   $rkwadraat = pow( $r, 2  );
5   $uitkomst = (  sin(  $rkwadraat ) * cos( $opp )  )/  tan(  pow( $r,  5 )   );
6 };
```

Sommige functies retourneren een waarde

- Invoer: er gaan nul of meer waarden in, dat kunnen ook variabelen zijn
- Verwerking: de functie verwerkt die invoer
- Returnwaarde: de functie “berekent” een returnwaarde en stuurt die waarde terug naar de aanroep.
- Deze functies zijn *altijd* onderdeel van een expressie:
  - Ze staan aan de *rechterkant* van het =-teken
  - Ze maken deel uit van een conditie

```
1 if  (is_numeric( $_POST['r'] )) {
2   $r  = $_POST['r'];
3   $opp = 2 * pi()  * $r;
4   $rkwadraat = pow( $r, 2  );
5   $uitkomst = (  sin(  $rkwadraat ) * cos( $opp )  )/  tan(  pow( $r,  5 )   );
6 };
```

Sommige functies retourneren *geen* waarde

- Invoer: er gaan nul of meer waarden in, dat kunnen ook variabelen zijn
- Verwerking: de functie verwerkt die invoer
- Effect: de functie doet iets met zijn omgeving: tekst afdrukken, praten met een database, een instelling veranderen, ...
- Deze functies staan op zichzelf en vormen een enkel PHP statement.
  - Ze maken *nooit* onderdeel uit van een expressie
  - Ze maken *nooit* onderdeel uit van een conditie

```
1 $var = "hallo"
2 var_dump( $var );
3 exit ();
```

Sommige functies retourneren *geen* waarde

- Invoer: er gaan nul of meer waarden in, dat kunnen ook variabelen zijn
- Verwerking: de functie verwerkt die invoer
- Effect: de functie doet iets met zijn omgeving: tekst afdrukken, praten met een database, een instelling veranderen, ...
- Deze functies staan op zichzelf en vormen een enkel PHP statement.
  - Ze maken *nooit* onderdeel uit van een expressie
  - Ze maken *nooit* onderdeel uit van een conditie

```
1 $var = "hallo"
2 var_dump( $var );
3 exit ();
```

Zelf functies definiëren

functiedefinitie

```
1 function functie_naam( $arg_1, $arg_2, ..., $arg_n ) {
2    // doe iets met de argumenten 1 tot en met n en bereken een waarde
3    ...
4    // en retourneer die waarde
5   return waarde;
6 };
```

- Nul of meer argumenten, maar altijd (en)
- Meer dan een return-statement mogelijk
- Als de functie niets retourneert, dan is er *geen* return-statement.

Zelf functies definiëren

functiedefinitie

```
1 function functie_naam( $arg_1, $arg_2, ..., $arg_n ) {
2    // doe iets met de argumenten 1 tot en met n en bereken een waarde
3     ...
4    // en retourneer die waarde
5    return waarde;
6 };
```

- Nul of meer argumenten, maar altijd (en)
- Meer dan een return-statement mogelijk
- Als de functie niets retourneert, dan is er *geen* return-statement.

Voorbeelden (I)

een functie om “Hallo!” te zeggen: geen argumenten, geen returnwaarde

```
1 function  hallo ()  {
2    echo "Hallo!";
3 };
```

een functie om iemand bij naam te begroeten: een argument, de naam en geen returnwaarde

```
1 function  hallo2(  $naam ) {
2    echo "Hallo $naam!";
3 };
```

Voorbeelden (II)

voorbeeldscript

```
1  <?php
2  function  hallo ()  {
3     echo "Hallo!";
4  };
5  function  hallo2(  $naam ) {
6     echo "Hallo $naam!";
7  };
8
9  hallo ();
10 hallo ();
11 hallo2(  "Maria" );
12 $naam = "Jan";
13 hallo2(  $naam );
14 ?>
```

verwerking en uitvoer

PHP springt naar regel 2 en voert regel 2–4 uit: PHP voert de functie hallo() uit en na uitvoer springt PHP weer terug naar regel 9.

Hallo!

Voorbeelden (II)

voorbeeldscript

```
1  <?php
2  function  hallo ()  {
3     echo "Hallo!";
4  };
5  function  hallo2(  $naam ) {
6     echo "Hallo $naam!";
7  };
8
9  hallo ();
   hallo ();
11 hallo2(  "Maria" );
12 $naam = "Jan";
13 hallo2(  $naam );
14 ?>
```

verwerking en uitvoer

PHP springt naar regel 2 en voert regel 2–4 uit: PHP voert de functie hallo() uit en na uitvoer springt PHP weer terug naar regel 10.

Hallo!
Hallo!

Voorbeelden (II)

voorbeeldscript

```
1  <?php
2  function  hallo ()  {
3    echo "Hallo!";
4  };
5  function  hallo2(  $naam ) {
6    echo "Hallo $naam!";
7  };
8
9  hallo ();
10 hallo ();
   hallo2(  "Maria" );
12 $naam = "Jan";
13 hallo2(  $naam );
14 ?>
```

uitvoer

PHP springt naar regel 5 en voert regel 5–7 uit: PHP voert de functie hallo2() uit. Van regel 11 neemt PHP de waarde *"Maria"* mee en kent dat *automatisch* toe aan de parameter $naam van de functie hallo2(). Na aanroep van een functie hebben alle parameters dus een lokale waarde.

Hallo!
Hallo!
Hallo Maria!

Voorbeelden (II)

voorbeeldscript

```
1  <?php
2  function  hallo ()  {
3     echo "Hallo!";
4  };
5  function  hallo2(  $naam ) {
6     echo "Hallo $naam!";
7  };
8
9  hallo ();
10 hallo ();
11 hallo2(  "Maria" );
   $naam = "Jan";
13 hallo2(  $naam );
14 ?>
```

uitvoer

Hallo!
Hallo!
Hallo Maria!

Voorbeelden (II)

voorbeeldscript

```
1  <?php
2  function  hallo ()  {
3    echo "Hallo!";
4  };
5  function  hallo2(  $naam ) {
6    echo "Hallo $naam!";
7  };
8
9  hallo ();
10 hallo ();
11 hallo2(  "Maria" );
12 $naam = "Jan";
   hallo2(  $naam );
14 ?>
```

uitvoer

PHP springt naar regel 5 en voert regel 5–7 uit: PHP voert de functie hallo2() uit. Van regel 11 neemt PHP de *waarde* van de variabele $naam mee (*"Jan"*) en kent dat *automatisch* toe aan de parameter $naam van de functie hallo2().

Hallo!
Hallo!
Hallo Maria!
Hallo Jan!

Voorbeelden (III)

een functie om het kwadraat van een getal te berekenen:, een argument en een returnwaarde

```
1 function kwadraat( $x ) {
2   return $x * $x;
3 };
```

een functie om het gemiddelde van een array met getallen te berekenen

```
1 function array_gemiddelde( $array ) {
2   $totaal = 0;
3   foreach( $array as $getal ) {
4     $totaal += $getal;
5   };
6   return $totaal / count( $array );
7 };
```

Voorbeelden (IV)

voorbeeldscript

```
1  <?php
2  function kwadraat( $x ) {
3    return $x * $x;
4  };
5  function array_gemiddelde( $array ) {
6    $totaal = 0;
7    foreach( $array as $getal ) {
8      $totaal += $getal;
9    };
10   return $totaal / count( $array );
11 };
12 $getal = kwadraat( 4 );
13 echo $getal;
14 $getal = kwadraat( $getal );
15 echo $getal;
16 $arr = array( 4, 2, 56, 2, 45, 45);
17 $arr[0] = kwadraat( $arr[0] );
18 echo array_gemiddelde( $arr );
19 ?>
```

uitvoer

```
16
256
27.66666667
```

Voorbeelden (IV)

voorbeeldscript

```
1  <?php
2  function kwadraat( $x ) {
3    return $x * $x;
4  };
5  function array_gemiddelde( $array ) {
6    $totaal = 0;
7    foreach( $array as $getal ) {
8      $totaal += $getal;
9    };
10   return $totaal / count( $array );
11 };
12 $getal = kwadraat( 4 );
13 echo $getal;
14 $getal = kwadraat( $getal );
15 echo $getal;
16 $arr = array( 4, 2, 56, 2, 45, 45);
17 $arr[0] = kwadraat( $arr[0] );
18 echo array_gemiddelde( $arr );
19 ?>
```

uitvoer

```
16
256
27.66666667
```

Waarom functies?

- Lange code? Hak je code op in logische kleinere stukken met functies
- Herhaal je veel code? Introduceer er een functie voor
- Gebruik je in verschillende PHP scripts vaak dezelfde functies? Stop ze in een apart bestand: *bibliotheek*
- Op de toets word je gevraagd om functiedefinities in te vullen.
- Een voorbeeld

Waarom functies?

- Lange code? Hak je code op in logische kleinere stukken met functies
- Herhaal je veel code? Introduceer er een functie voor
- Gebruik je in verschillende PHP scripts vaak dezelfde functies? Stop ze in een apart bestand: *bibliotheek*
- Op de toets word je gevraagd om functiedefinities in te vullen.
- Een voorbeeld

Waarom functies?

- Lange code? Hak je code op in logische kleinere stukken met functies
- Herhaal je veel code? Introduceer er een functie voor
- Gebruik je in verschillende PHP scripts vaak dezelfde functies? Stop ze in een apart bestand: *bibliotheek*
- Op de toets word je gevraagd om functiedefinities in te vullen.
- Een voorbeeld