6.4 Generieke Functies

Let op: voor deze opgave moet de opgave Controle Bankrekeningnummer worden uitgevoerd.

We hebben al geoefend met functies, maar we gaan nog een keer theoretisch kijken naar functies; wanneer gebruik je functies, wat zij de voordelen en waaraan voldoet een goede functie?

We hebben het her over functies, maar alles wat voor functies geldt ook voor methods in een class.

Wat is een (goede) functies

Een functie is een stukje code dat **hergebruikt** kan worden. Dat betekent dat functies generiek zijn. Dat betekent dat een goede functies een toepassing heeft die algemeen toepasbaar is.

Als je in grotere teams samenwerkt aan een groot project dan kun je verschillende developers laten werken aan verschillende functies. Als je de input en output goed beschrijft, dan hoef je niet precies te weten hoe de functie (intern) werkt. Je kunt het gewoon als een stukje code gebruiken. Je moet dan wel zorgen dat de functie algemeen toepasbaar is; generiek dus.

Van een functie is het heel belangrijk wat de input en de output is. Als je de input (parameters) en de output (return value) goed beschrijft, dan hoef je niet te weten hoe de functie (intern) werkt om hem te kunnen gebruiken.

Voorbeeld

Laten we eens kijken naar de (ingebouwde) PHP-functie rand().

```
rand ( int $min , int $max ) : int


Parameters
$min: De minimale waarde
$max: De maximale waarde


Return Values
Een willekeurig getal tussen min en max.
```

(bron: https://www.php.net/manual/en/function.rand.php)

Zoals je kan zien geef je twee waardes mee, beide een integer. De eerste integer bepaald de minimale waarde, de tweede de maximale. En de return value is een willekeurig getal (ook int) tussen de min en max waarde.

Deze functie is generiek want die kan voor heel veel doeleinden worden gebruikt. Met de parameters kun je als het ware het gedrag van de functie sturen en bepaal je wat de functie doet. Wil je een dobbelsteen simuleren, of wil je een getal tussen 0 en 100, of wil je kop/munt spelen, dat kan allemaal met deze functie.

Een voorbeeld van een minder generieke functie zou zijn:

```
rand () : int

Parameters
Geen

Return Values
Een willekeurig getal tussen 1 en 10.
```

Om met deze functie een dobbelsteen te simuleren wordt een stuk lastiger.

Een voorbeeld van een nog minder generieke functie zou zijn:

```
rand ()

Parameters
Geen

Return Values
Geen

De functie print een getal tussen 0 en 10
```

In dit voorbeeld is er geen parameter en geen return value. De functie print (echo) een getal tussen 0 en 10 want je kunt bijvoorbeeld niet meer rekenen met de uitkomst van deze functie.

Hoe maken we functies generiek?

Daar zijn geen kei-harde regels voor, maar er is wel een aantal zaken waar je aan kan denken:

1. Is de output (return value) van een functie altijd hetzelfde?

Als dat zo is, dan is de functie waarschijnlijk niet heel erg generiek.

2. Zit er een echo/print in een functie?

Met een echo ligt het formaat van de output vast, bijvoorbeeld de taal. Ook kun je met het resultaat niet meer verder rekenen. Een echo kan er dus ook voor zorgen dat een functie minder generiek wordt.

3. Is de output generiek?

Is de output algemeen toepasbaar? Stel je hebt een functie die het gemiddelde berekend en je return ziet er als volgt uit:

```
return "Gemiddelde is: ".$som/$aantal;
```

De return value is nu taalafhankelijk en met de output is het lastig verder rekenen. Stel je wilt het gemiddelde vermenigvuldigen met 2, hoe zou je dat dan doen? Het is generieker om alleen een getal te returnen.

Sommige functies hebben als output een ja of nee. Bijvoorbeeld een functie die terug geeft of je gemiddelde een voldoende is. Als dat het geval is gebruik je bij voorkeur als return value true of false.

Je kunt dan bij het aanroepen van de functie gelijk een if statement maken:

```
if ( isgemiddeldeVoldoende($array) ) {
echo "Je hebt een voldoende!";
} else ...
```

4. Is de output afhankelijk van de input?

Bepaal je met de input van de functie de output? Stuur je als het ware wat de functie doet met input? Als dat zo is dan maakt dat de functie meer generiek.

Een generieke functie kan dus op meerdere plaatsen in je code worden gebruikt. Je kunt ook een library maken met functies en die dan met anderen delen. Dat is wel erg generiek en als dat gebeurt dan wil je ook je functie robuust is en niet zomaar verkeerde resultaten geeft.

Hoe maken we een functie robuust?

4. Wordt er gecontroleerd of de parameter een goede waarde hebben?

Een generieke functie heeft parameters. Er zijn gevallen dat niet alle parameters tot een goed resultaat leiden. Als je bijvoorbeeld en functie hebt die een getal afrond op $n decimalen dan wil je dat $n groter of gelijk aan nul is. Dit zorgt ervoor dat de gebruiker van de functie fouten kan maken en dat die dan netjes worden afgevangen of in ieder geval niet leiden tot andere fouten.

Video

In dit voorbeeld hieronder wordt nog een keer gekeken naar de functie waarbij een bankrekeningnummer wordt gecontroleerd. De functie wordt generiek gemaakt door de return value aan te passen.

https://www.youtube.com/embed/Z7OKbmQYPgY

Voorbeelden

Met deze functie 'gooien' we met twee dobbelstenen.

```
function dobbel() {
    $steen1 = rand(1,6);
    $steen2 = rand(1,6);
    echo $steen1 + $steen2;
}
```

Kijken we naar de 3 richtlijnen dan zien we dat er een echo in de functie zit. We kunnen de functie dus generieker maken.

```
function dobbel() {
    $steen1 = rand(1,6);
    $steen2 = rand(1,6);
    return $steen1 + $steen2;
}
```

Nu is de functie al generieker, maar het kan nog beter want wat als we met 3 of 4 dobbelstenen zouden willen gooien; dat kan nu niet?

```
function dobbel($aantal) {
    $totaal = 0;
    for($i=0; $i<$aantal; $i++){
        $gooi=rand(1,6);
            $totaal = $totaal + $gooi;
        }
    return $totaal;
}
```

Nu hebben we een functie waarbij de gebruiker kan aangeven met hoeveel dobbelstenen hij gaat gooien. Een goede functie zorgt ervoor dat er niet fout kan gaan. Een goede functie is robuust. Is de bovenstaande functie robuust? Wat gebeurt er als je bijvoorbeeld -1 als parameter meegeeft? De functie loopt niet vast of zo en hij returned 0. Dat is (toevallig) goed, een 0 of een -1 als return value duidt soms op een fout.

Opgave 1

De function stringGelijk($string1, $string2) vergelijkt twee strings onafhankelijk van het hoofdletter gebruik. Dus "Appel" is gelijk aan "appel".

Is de functie generiek? Kun je hem misschien generieker maken? Pas de code aan.

```
<?php

function stringGelijk($string1, $string2) {

  if ( strtoupper($string1) == strtoupper($string2) ) {
    return "Gelijk";
  } else {
    return "Ongelijk";
  }

}

echo stringGelijk("Hallo", "hallo");
```

Opgave 2

De function berekenGemiddelde() berekend het gemiddeld van een array.

Pas de code aan, zodat die generieker wordt.

```
function berekenGemiddelde() {
$aantal = 0;
    $som = 0;
$array = [5,4,7,6,5,6,5,7,6,7,8,3,4,7,6,7];
    for($i=0; $i<count($array); $i++) {
    $som = $som + $array[$i];
        $aantal++;
    }
```

```
    return $som/$aantal;
}
```

--

Revision #8
Created 2020-04-29 16:38:28 UTC by Max
Updated 2020-05-10 09:13:24 UTC by Max