Objecten - extra uitleg

Deze pagina is een copy van:
http://www.sitemasters.be/tutorials/1/1/607/PHP/OOP_(Object_Oriented_Programming)

In deze tutorial hoop ik uitleg te geven over hoe met een begin kan maken met het object georiënteerd programmeren (OOP). Ik ga ervan uit dat je al wat basiskennis hebt van PHP en dat je gebruik maakt van PHP 5.3 of hoger.

Objecten zijn “toewijzingen” van classes. Een class is een soort van 'blauwdruk', waarin je de logica van het object begint. Dit klinkt lastig, maar ik zal het uitleggen met een voorbeeld. Ik zal het voorbeeld nemen van een user-module. In deze module gaan we een aantal dingen stoppen. We gaan de mogelijkheid geven om meerdere eigenschappen op te geven en willen de mogelijkheid geven voor een user om iets te zeggen.

In OOP zou dit er zo uitzien:

```
<?php
Class User {
    public $name;
    public $age;

    public function __construct($name,$age) {
        $this->name = $name;
        $this->age = $age;
    }

    public function say($message) {
        echo $this->name . ': ' . $message . '<br />';
    }

    public function sayAge() {
        $this->say('Ik ben ' . $this->age . ' jaar.');
    }
}
```

```
$piet = new User('Piet',23);
$klaas = new User('Klaas',26);
$bert = new User('Bert',29);

$piet->say('Bert, alles goed?');
$bert->say('Jawel man!');
$klaas->say('Hoe oud ben jij Piet?');
$piet->sayAge();
?>
```

Er wordt een Class 'User' aangemaakt. In deze class geef je allerlei mogelijkheden (in dit geval om een naam/leeftijd op te geven, om iets te willekeurigs te zeggen en om je leeftijd te zeggen.

Onder de class maak ik drie objecten aan ($piet, $klaas en $bert). Op het moment dat ik zeg 'new User', weet PHP dat ik een class bedoel en gaat hij op zoek naar de class. Omdat ik achter newUser ('Piet',21) heb gezet gaat hij op zoek naar de __construct method (die is de standaard 'init' functie) en hij verwacht 2 gegevens (naam en leeftijd). Hierdoor komt de class eigenlijk tot leven en worden zijn alle variabelen/functies tot mijn beschikking bij Piet. Daarna doe ik hetzelfde voor de andere personen en uiteindelijk heb ik dus drie gebruikers.

Nu kan ik via het object ($piet, $berg en $klaas) de methodes aanroepen en ook de variabelen. Als ik $piet->say('iets'); opgeef betekent dit dat ik de 'say' method aanroep van het object piet. Piet gaat hierdoor praten en je ziet zijn naam staan, omdat het object de naam heeft gekregen bij het initialiseren (__construct).

Ik zei ook dat je eventueel de variabelen direct aan zou kunnen roepen. Dit kan door het object aan te roepen en de variabelenaam erachter. Als ik de leeftijd van piet wil weten, kan ik deze zo ophalen:

`$piet->age;`

Bij de methode 'sayAge' zien we iets anders wat heel belangrijk is, en dat is een speciale variabele '$this'. In deze variabele zitten alle variabelen en methodes van het object. Dus als je in het object $piet zit en je gebruikt $this, zit daar alles in van het object piet. Dus je zou kunnen doen $this->say(), $this->name, $this->age. Dit geeft je dus de mogelijkheid om vanuit één method, andere methods aan te roepen die binnen deze class zitten.

Voor ingevulde variabelen/argumenten
Het is de mogelijkheid (net als bij normale functies) om voor ingevulde variabelen op te geven bij de methods. Deze hoef je dan niet verplicht mee te geven als je de method aanroept.

```
<?php
```

```
public function setGender($gender = 'M') {
    $this->gender = $gender;
}


?>
```

Als ik in dit voorbeeld $piet->setGender(); zou opgeven, dan wordt zijn gender (geslacht) op 'M' gezet.

Revision #1
Created 26 September 2019 10:28:11 by Admin
Updated 26 September 2019 10:33:04 by Admin