PHP OOP

Applicatie / Media development jaar 2

Marcel Koningstein

INHOUDSOPGAVE

Inleiding

Tijdens de lessen Development zal je in aanraking komen met OOP, hierin krijg je uitleg over wat OOP is en leer je de basis principes van het werken met OOP. De komende weken zal je verschillende opdrachten uitvoeren om deze basis onder de knie te krijgen.

Aan het einde van deze lessen kan/weet je het volgende:

- Je weet wat OOP is
- Je weet wat een Class is
- Je weet wat een Object is
- Je weet wat properties zijn
- Je weet wat visibility is
- Je weet wat een construct is
- Je weet wat overerving is
- Je weet wat een abstracte class is
- Je weet wat een interface is
- Je weet wat een namespace is
- Je kan alle bovenstaande OOP principes toepassen

De opdrachten vinden plaats in 6 weken, waarin tevens op het eind een schriftelijke toets zal plaatsvinden.

Eindcijfer van OOP Basis zal gebaseerd zijn op:

- Maken van de OOP opdrachten met de aangemaakte Github repositories
- Uiteindelijk opgeleverde opdrachten in natschool
- Cijfer van de schriftelijke toets

OOP opdrachten

Opdracht 1a: OOP Huizen

Nodige kennis: class, property, visability

Voor een woonwijk moeten een aantal huizen gebouwd worden.
Per huis kan er van alles verschillen. De huizen hebben allemaal een
aantal verdiepingen, aantal kamers, een breedte, een hoogte en een diepte. De prijs hangt af van het aantal kubieke meters.

Maak 3 huizen aan en print deze uit met alle details met de prijs.

Opdracht 1b: Herberekening huisprijs

Nodige kennis: class, property, visability, array

Het blijkt dat de berekening van de prijs bij opdracht 1a niet goed is gegaan. We moeten van elke ruimte weten hoe groot ze zijn om de exacte prijs te bepalen van het huis.

Dit betekent dat een kamer een aparte class moet gaan krijgen, waarbij we de volume gaan bepalen met de lengte, hoogte en breedte.

Als we een kamer aanmaken, moeten we er wel voor zorgen dat de kamer aan een huis is gekoppeld. Dit doen we door het object van de kamer in een array te zetten.

Om de prijs te bepalen, zal eerst de grootte van elke kamer berekend moeten worden. De grootte van een kamer hoort bij de kamer, en zal dus in die class moeten gebeuren.

De totale grootte van het huis hoort bij het huis. Je zal dus de array van kamers moeten gebruiken om de totale grootte te bepalen. (loop door de array heen)

Als de totale grootte van het huis is bepaald, kan de prijs worden bepaald.

Opdracht 2: Drie op een rij

Nodige kennis: class, property, visability, inherritance

Bij deze opdracht gaan we een begin maken van een spel, namelijk drie op een rij. Voor deze versie van drie op een rij gaan we gebruik maken van allerlei soorten figuren:

- Vierkant
- Driehoek
- Rechthoek
- Cirkel

Elk figuur kan een kleur hebben: Rood, groen, blauw, geel, oranje of paars.

Stap 1:

Maak eerst de parent class aan: Figuur
Bepaal de algemene eigenschappen van een figuur.

Stap 2:

Maak de child classes aan voor de soorten. Kijk goed wat je nodig hebt om het figuur te tekenen.

Stap 3:

Zorg dat elk figuur op je scherm staat (html + css)

Opdracht 3 – Producten (abstract)

Voor een nieuwe webshop moeten er verschillende producten op de site komen te staan. Het gaat om producten als muziek dvd's, films en games.

- Een muziek dvd heeft verschillende nummers erop staan. Het zou handig zijn als we de namen van alle nummers op de site hebben staan.
- Een film is van een bepaalde kwaliteit, zoals dvd en blueray.
- Games hebben een bepaald genre en minimale hardware eisen

Stap 1:

Maak voor deze opdracht een abstracte class Product. Een product heeft in ieder geval de volgende eigenschappen: naam, inkoopprijs, btw, omschrijving.
Een product heeft ook een abstracte methode die alle product informatie terug geeft.

Stap 2:

Maak de child classes Music, Film en Game. Zorg dat de verschillende soort producten netjes hun eigen eigenschappen hebben.

Stap 3:

Alle producten moeten in een lijst komen. In de lijst moet minimaal staan:

- Naam van product
- Categorie (music, film, game)
- Verkoopprijs (inkoopprijs + winst + btw)

Maak hiervoor een class ProductList aan.

Opdracht 4 – Ziekenhuis

Een ziekenhuis wilt zijn patiënten en medewerkers straks kunnen beheren. Om dit te kunnen beheren is het de bedoeling dat deze patiënten en medewerkers een apart object krijgen. De medewerkers zijn ook nog eens verdeeld in: doctoren en assistentes. De assistentes en doctoren hebben allemaal een apart salaris dat zij verdienen en een patiënt moet geld betalen voor zijn behandeling. Aangezien alle objecten in dit geval een persoon is, is het goed om hiervoor een apart object te maken. Een persoon heeft zelf nog verschillende eigenschappen zoals: kleur ogen, kleur haar, lengte, gewicht en een rol. Verder moet het mogelijk zijn om afspraken tussen verschillende rollen.

De salaris van staff zal berekent worden door de uren die ze afspraken hebben * uurloon. Niet bij alle afspraken zijn de Nurses nodig, dus deze hebben een werkweek van 40 uur met een vast loon, waarbij het geld van afspraken als bonus erbij komt. Doctoren krijgen alleen per afspraak betaald.

Stap 1:

De opdracht is om een abstracte class Person aan te maken. In deze class moet een abstracte methode zitten die de rol bepaald. Zorg dat de properties zoveel mogelijk gesloten zijn (private). Maak Setters en Getters aan om bijvoorbeeld apart de naam straks op te kunnen vragen etc.

Stap 2:
Maak de child class Patient aan. Van patient moet je een object kunnen maken, dus een gewone class met overerving van Person. Zorg dat de abstracte methode van Person wordt uitgewerkt in Patient.

Stap 3:
Maak de child class Staff aan die overerft van Person. Staff zal straks worden onderverdeeld in Doctor en Nurse, dus class zal abstract worden. In deze class zal een abstracte methode komen om salaris te berekenen.

Stap 4:
Maak de child class Doctor aan die overerft van Staff. Werk alle methodes uit.

Stap 5:
Maak de child class Nurse aan die overerft van Staff. Werk alle methodes uit.

Stap 6:
Maak de class Appointment aan. Een appointment is tussen een Doctor en Patient, optioneel met Nurse. Werk het optionele gedeelte pas later uit.
Bij de afspraak heb je een begin en eind tijd. Hiervan moet je de duur uitrekenen om de kosten (loon van Doctor en bonus van Nurse) te bepalen

Tips!

- Probeer bij het maken van de afspraken static te gebruiken

Opdracht 5 – Schooluitje

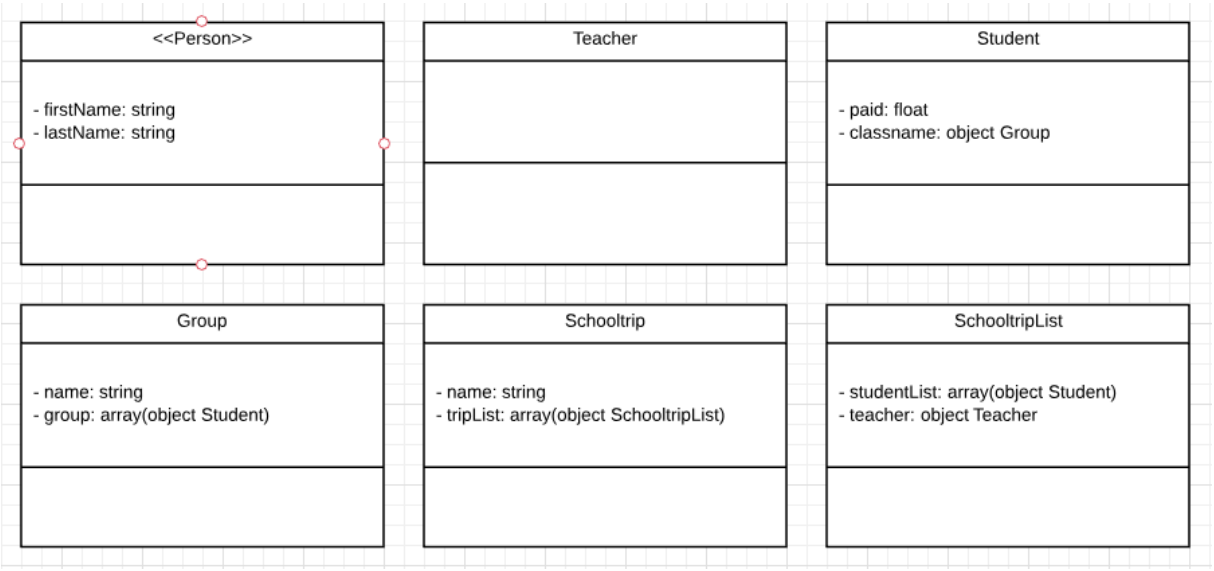
Bij de opleiding applicatie en media willen de leerlingen graag een schooluitje organiseren. Het idee was om naar een pretpark te gaan. Hiervoor zal geld ingezameld moeten worden van elke leerling.

Hierbij willen we ook graag overzicht hoeveel procent van een klas zich heeft opgegeven voor het uitje, en of deze al hiervoor heeft betaald. Verder moet er een totaal komen van het ingezamelde bedrag per klas, maar ook het ingezamelde bedrag in totaal.

Bij een schooluitje moet er ook begeleiding zijn van een docent. Per 25 betalende leerlingen kan 1 docent mee. Zodra er plek is voor een docent kan deze ook op de lijst komen die meegaan met het uitje. De docent hoeft niet te betalen, en wordt ook niet meegenomen in percentages.

Het eindresultaat moet een lijst zijn, waarbij per 25 leerlingen 1 docent staat.
Op de lijst is zichtbaar in welke klas ze zitten en of ze betaald hebben. Niet alle leerlingen van een klas gaan mee.

In deze opdracht zal je te maken krijgen met inherritance, abstract en static.
Gebruik onderstaand schema. Werk het UML schema verder uit en ga daarna pas beginnen met programmeren.



Opdracht 6 - Recepten

Voor deze opdracht gaan we OOP gebruiken met een database. Hiervoor hebben we PDO nodig, wat je ook in het 1e jaar hebt gebruikt. Nu je de basis van OOP kent, zie je waarschijnlijk in dat je eigenlijk al in de 1e klas wat OOP dingen hebt gedaan. PDO is namelijk al OOP.

Het enige is, dat bij OOP het hergebruik van code erg belangrijk is. Zoals dat in de 1e klas wordt gebruikt is nog niet dat je het kan hergebruiken.

Hiervoor hebben we nog een klas nodig die alles doet met de database. De bedoeling is om methodes te maken voor een insert, update, delete en select. (bewaar de select voor het laatst)

Voor deze opdracht moet er een systeem gebouwd worden waarop mensen verschillende recepten kunnen lezen. Een recept bestaat uit een aantal dingen namelijk: Een omschrijving, ingrediënten, categorie, commentaar en een bereidingswijze. De recepten moeten in een database komen.