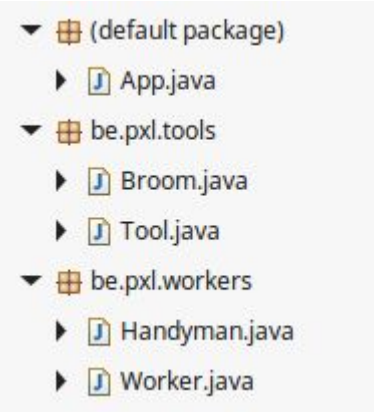
Oefeningen OOP in PHP

(1) Gegeven de onderstaande Java code.



be/pxl/tools/Tool.java

```
package be.pxl.tools;
public interface Tool {
     public void doSomething();
}
```

be/pxl/tools/Broom.java

```
package be.pxl.tools;
public class Broom implements Tool {
     public void doSomething() {
          System.out.println("Sweep");
     }
}
```

be/pxl/workers/Worker.java

```
package be.pxl.workers;
public interface Worker {
     public void work();
}
```

be/pxl/workers/Handyman.java

```
package be.pxl.workers;
import be.pxl.tools.Tool;
public class Handyman implements Worker {
     private Tool tool;

     public Handyman(Tool tool) {
          this.tool = tool;
     }

     public void work() {
          tool.doSomething();
     }
}
```

App.java

```
import be.pxl.tools.Broom;
import be.pxl.workers.Handyman;
public class App {
     public static void main(String[] args) {
          Broom broom = new Broom();
          Handyman henry = new Handyman(broom);
          henry.work();

     }
}
```

Vertaal deze code naar PHP. Zorg dat de onderstaande directory-structuur en namespaces gebruikt wordt:

- src/tool
- src/worker

Maak een autoloader via composer.

(2.a) Maak de klasse Date in het bestand src/Util/Date.php

Date heeft eigenschappen day, month, year.

Voorzie een constructor om een datum op volgende manieren te maken:
- zonder argumenten: datum = 1/1/2008
- met enkel een argument voor dag en maand: datum = dag/maand/2008
- met 3 argumenten

Denk er aan dat je maar één functie __construct kan voorzien in de klasse Date. Je kan hier werken met default arguments.

De datum moet in het formaat 1/1/2008 afgedrukt kunnen worden via de methode print().

Voorzie een extra methode printMonth() die de datum als volgt afdrukt: 1/jan/2008.

Maak hiervoor gebruik van een private static array $MONTHS.

Voorzie setters en getters voor dag, maand en jaar.

Maak in invoer.html een formulier om dag, maand en jaar in te geven.
In verwerk.php wordt een Date-object aangemaakt en via de methode printMonth afgedrukt.

(2.b) Vertrek van bovenstaande code. Zorg ervoor dat de constructor private is en voorzie een static functie make om Date-objecten aan te maken.

```
$date1=Date::make();
$date2=Date::make(1,2);
$date3=Date::make(1,2,2001);
```

Maak Date immutable. Verwijder de setters en voorzie de functies changeDay, changeMonth, changeYear

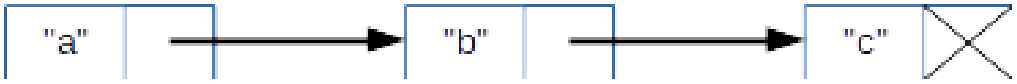
```
$date4=$date->changeDay(2);
```

(3)

Gegeven de klasse TextNode in het bestand TextNode.php in de directory textnode.

Een TextNode heeft als eigenschappen de variabele text en een verwijzing naar een volgend TextNode-object (nextNode). Op deze wijze wordt een keten van nodes gemaakt.

Een keten die bestaat uit tekstwaarden "a", "b", "c" kan voorgesteld worden als



Op de volgende bladzijde wordt de code van de klasse TextNode getoond.

De commentaar regels die beginnen met "// ..." duiden plaatsen aan waar je de code nog moet vervolledigen.

Zorg ervoor dat de namsepace textnode gebruikt wordt.

Je kan de code testen in App.php:

```
require(...);
$n=textnode\TextNode::makeNode("a");
$n->addNode("b");
$n->addNode("c");
$n->printAll();
```

```php
<?php
class TextNode
{
  private $nextNode;
  private $text;

  private function __construct($text)
  {
    // ... de eigenschap nextNode krijgt waarde null
    // ... de eigenschap text krijgt een waarde
  }

  public static function makeNode($text)
  {
    // ... roep de constructor aan en geef dit object terug
  }

  // de functie addNode voegt een TextNode toe op het einde van
  //een keten van nodes
  public function addNode($text)
  {
    // kijk of nextNode gelijk is aan null
    // indien ja: maak een nieuwe node aan en ken die toe aan
    //nextNode
    if($this->nextNode==null){
      $this->nextNode=self::makeNode($text);
    }
    else{
      // indien nee: roep de methode addNode aan op nextNode
      $this->nextNode->addNode($text);
    }
  }

  public function printAll()
  {
      print($this->text);
      if($this->nextNode !=null){
        $this->nextNode->printAll();
      }
  }

}
```

Pas de klasse aan, voeg de functie printTextNodeAt toe. Deze functie heeft $i als argument en print de TextNode op index $i uit de keten af.

```
$n=textnode\TextNode::makeNode("a");
$n->addNode("b");
$n->addNode("c");
$n->printElementAt(2);
```

Het aanroepen van de functie printElementAt op de laatste regel zorgt ervoor dat "c" afgedrukt wordt ("a" staat op index 0, "b" op index 1 en "c" op index 2).

Voor negatieve indices en voor indices die te hoog zijn wordt niets afgedrukt.

Extra oefening

Gegeven de onderstaande Java-code.

```
package be.pxl.util;

public class Color{
     public static Color BLACK = new Color(0x000000);
     public static Color WHITE = new Color(0xFFFFFF);
     public static Color RED = new Color(0xFF0000);
     public static Color GREEN = new Color(0xFF00FF);
     public static Color BLUE = new Color(0x0000FF);
     public static Color YELLOW = new Color(0xFFFF00);

     private int rgb;
     private Color(int rgb) {
          this.rgb=rgb;
     }

     public  int getRgb() {
          return rgb;
     }

     public String toString() {
          return Integer.toHexString(rgb);
     }

}
```

(Opmerking normaal zou je voor Color een enum gebruiken maar deze structuur bestaat niet in PHP).

Vertaal deze naar PHP. Maak de klasse ColouredPoint afgeleid van Point. Zorg dat je een ColouredPoint enkel kan aanmaken (maak de constructor private) via

```
$point = ColouredPoint::make(1,2, Color::RED);
```

Via

```
$point->print();
```

wordt de (x,y)-coördinaat en de rgb waarde afgedrukt.