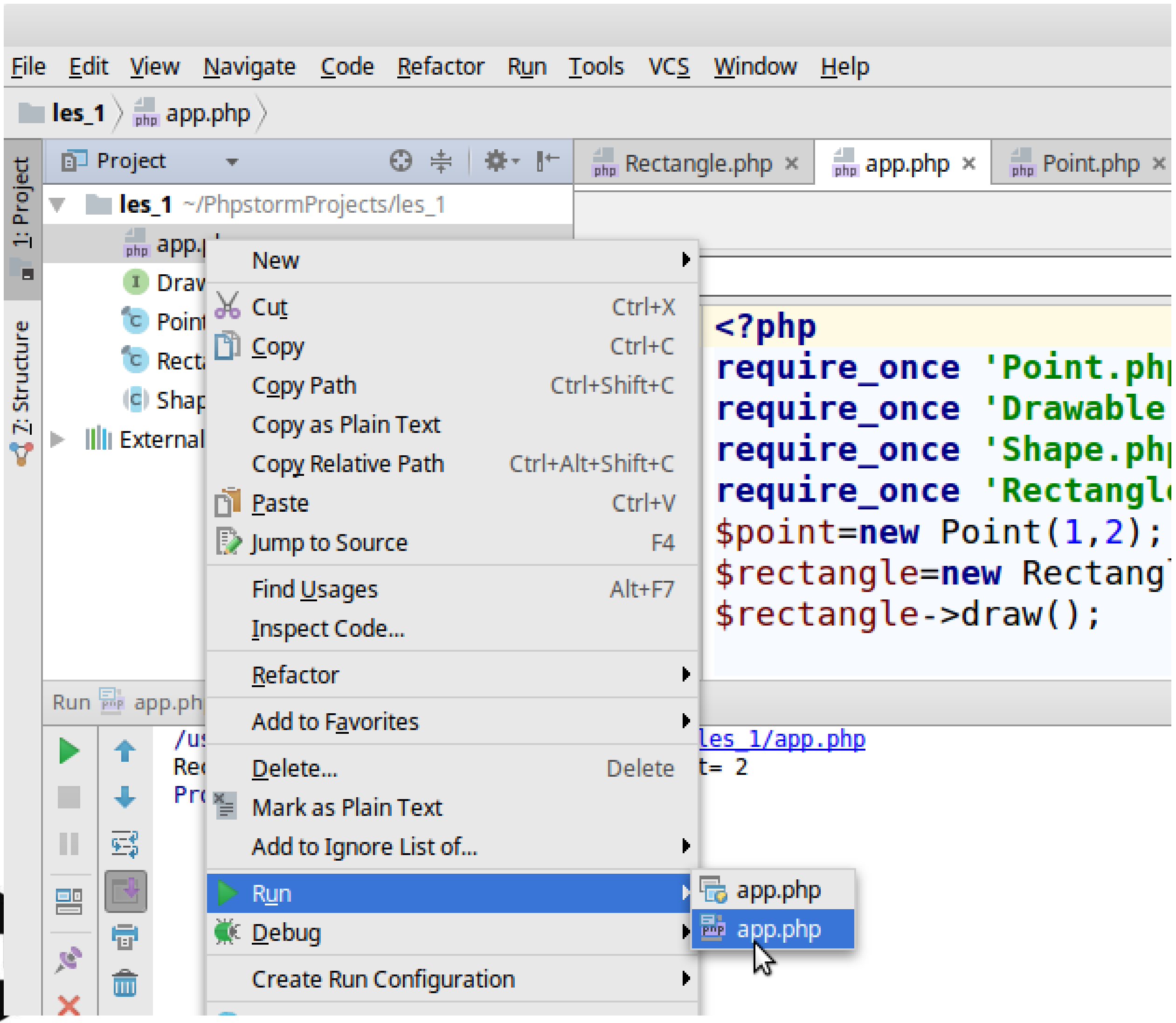
Web advanced

OOP

DE HOGESCHOOL
MET HET NETWERK

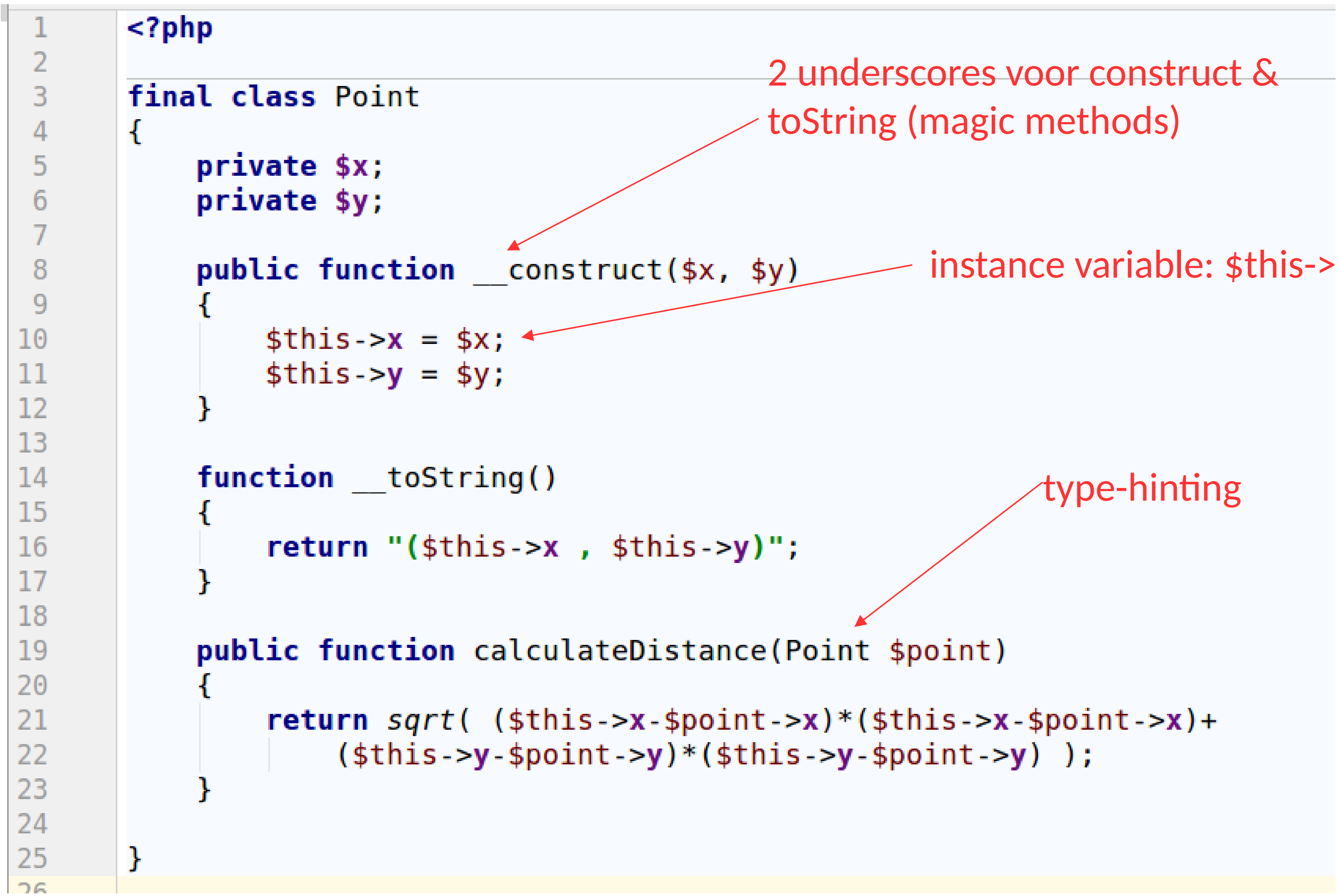
Hogeschool PXL – Dep. PXL-IT – Elfde-Liniestraat 26 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook

Software



```
<?php
require_once 'Point.ph
require_once 'Drawable
require_once 'Shape.ph
require_once 'Rectangl
$point=new Point(1,2);
$rectangle=new Rectang
$rectangle->draw();
```

Voorbeeld1: instance



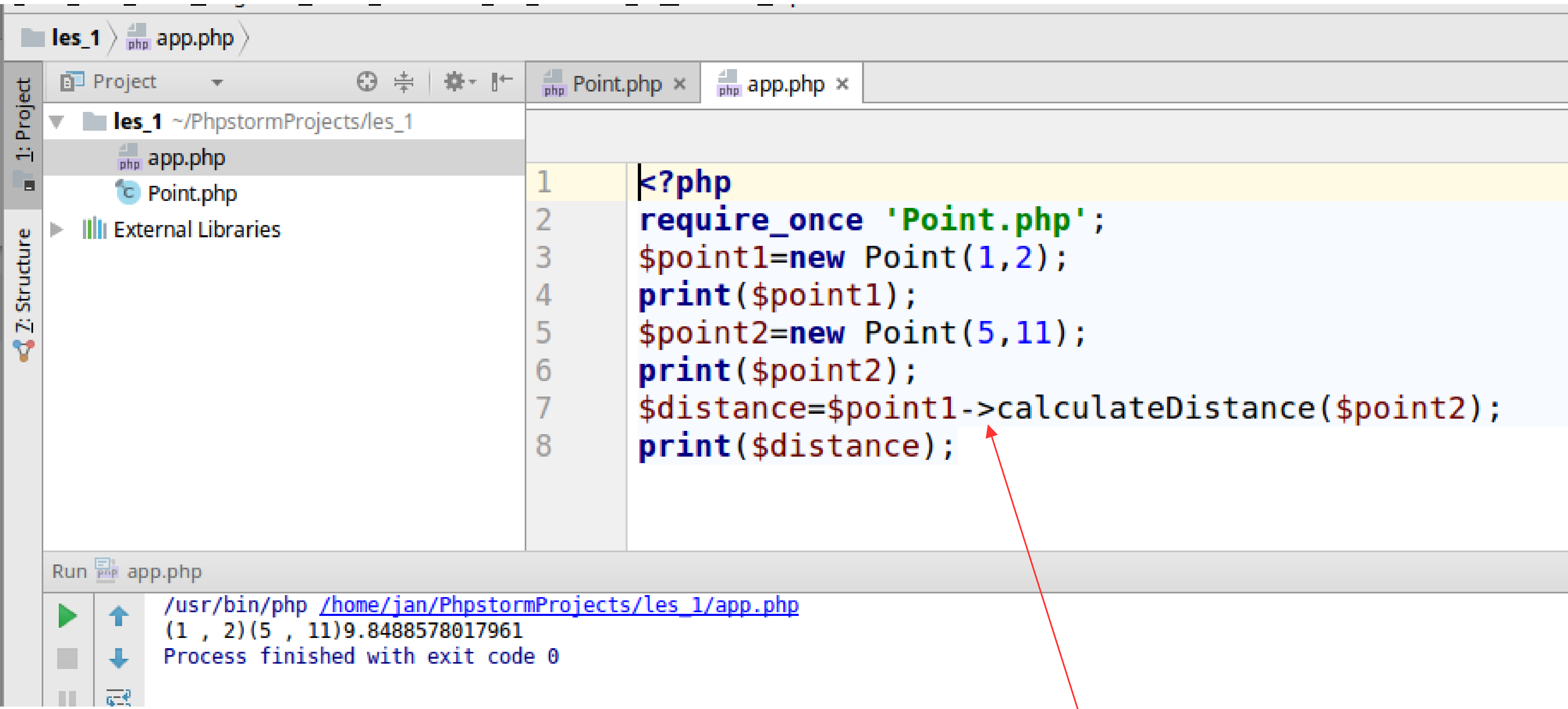
```php
<?php

final class Point
{
    private $x;
    private $y;

    public function __construct($x, $y)
    {
        $this->x = $x;
        $this->y = $y;
    }

    function __toString()
    {
        return "($this->x , $this->y)";
    }

    public function calculateDistance(Point $point)
    {
        return sqrt( ($this->x-$point->x)*($this->x-$point->x)+
            ($this->y-$point->y)*($this->y-$point->y) );
    }

}
```

2 underscores voor construct & toString (magic methods)

instance variable: $this->

type-hinting

Voorbeeld1: instance

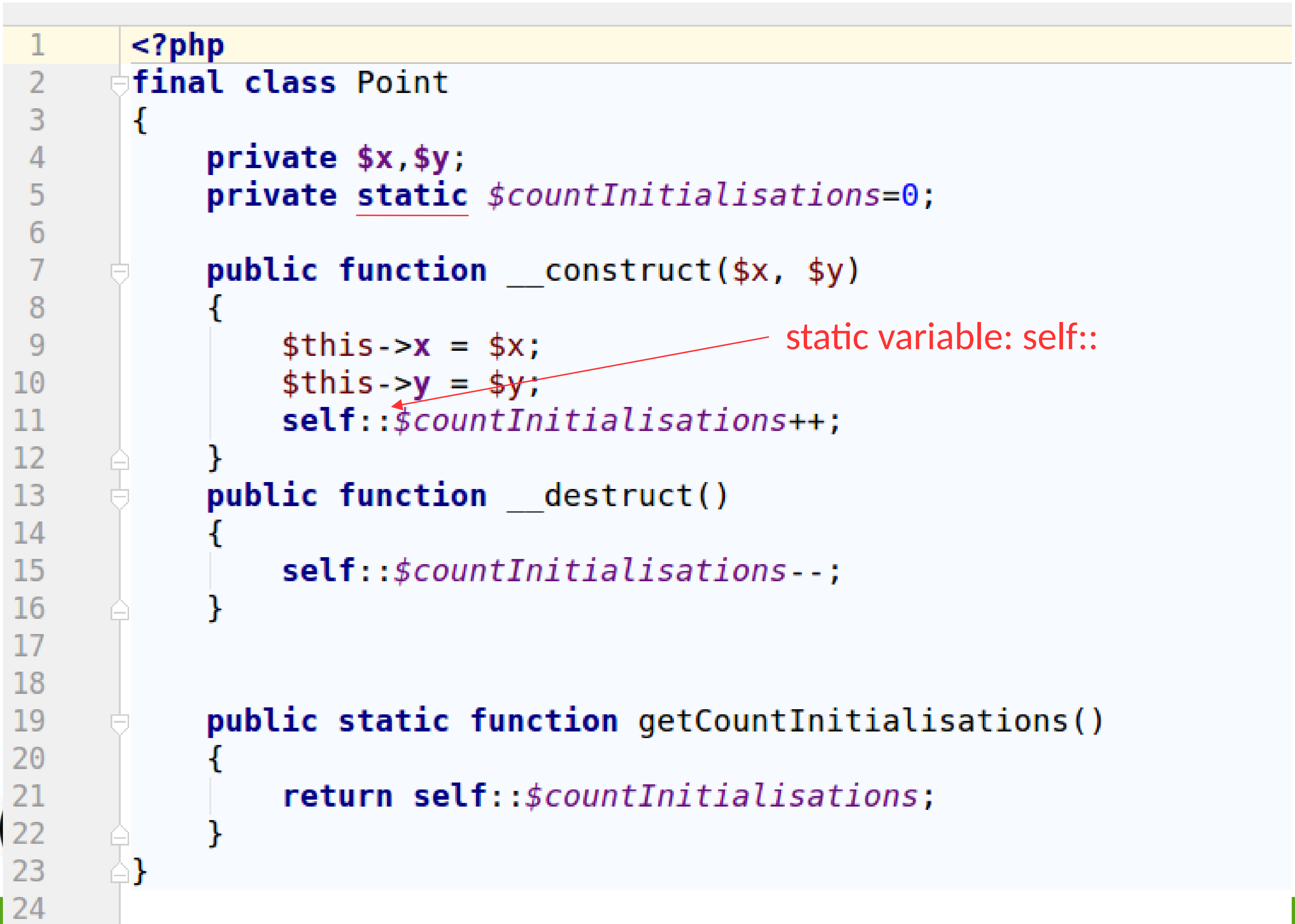


```php
<?php
require_once 'Point.php';
$point1=new Point(1,2);
print($point1);
$point2=new Point(5,11);
print($point2);
$distance=$point1->calculateDistance($point2);
print($distance);
```

```
/usr/bin/php /home/jan/PhpstormProjects/les_1/app.php
(1 , 2)(5 , 11)9.8488578017961
Process finished with exit code 0
```

instance method:
$object->

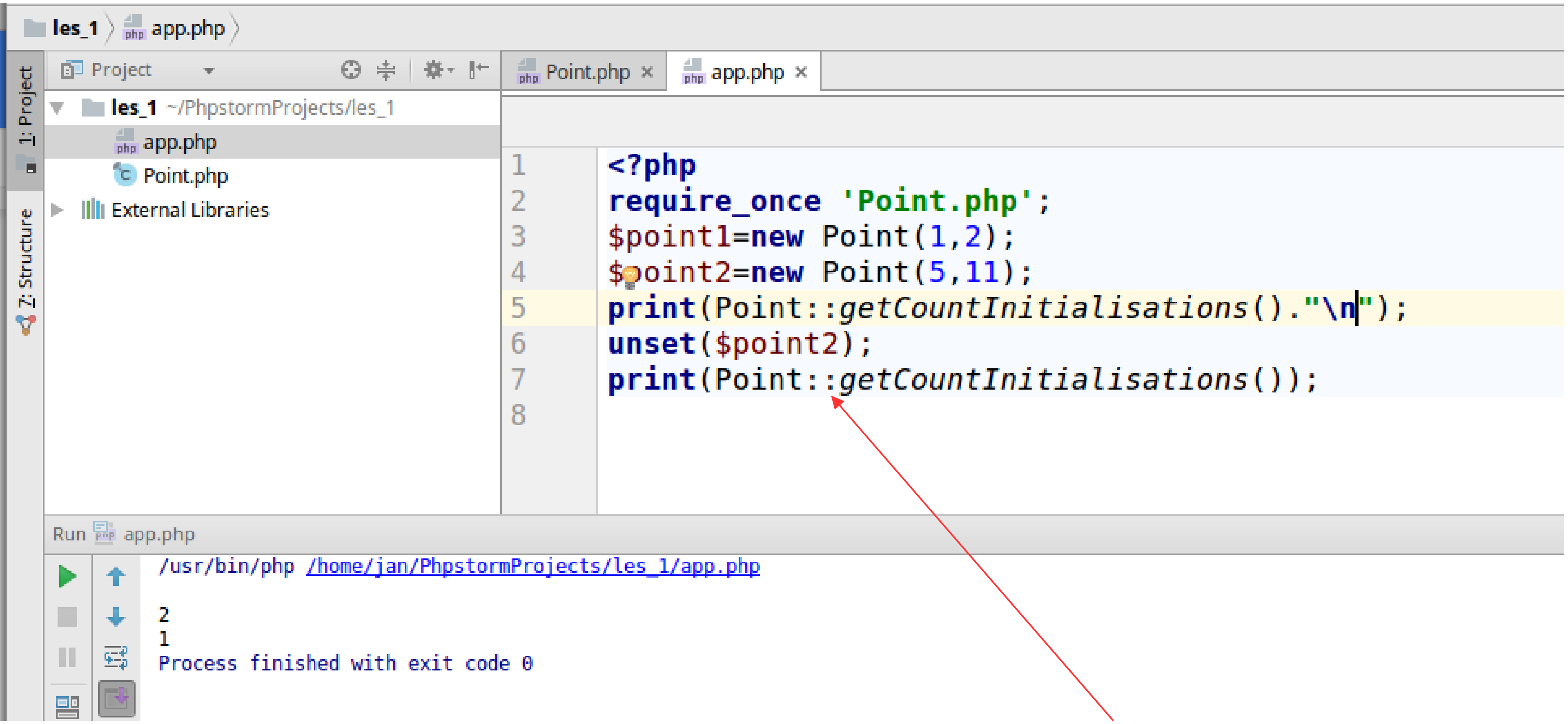
Voorbeeld2: static



```php
<?php
final class Point
{
    private $x,$y;
    private static $countInitialisations=0;

    public function __construct($x, $y)
    {
        $this->x = $x;
        $this->y = $y;
        self::$countInitialisations++;
    }
    public function __destruct()
    {
        self::$countInitialisations--;
    }


    public static function getCountInitialisations()
    {
        return self::$countInitialisations;
    }
}
```

static variable: self::

Voorbeeld2: static

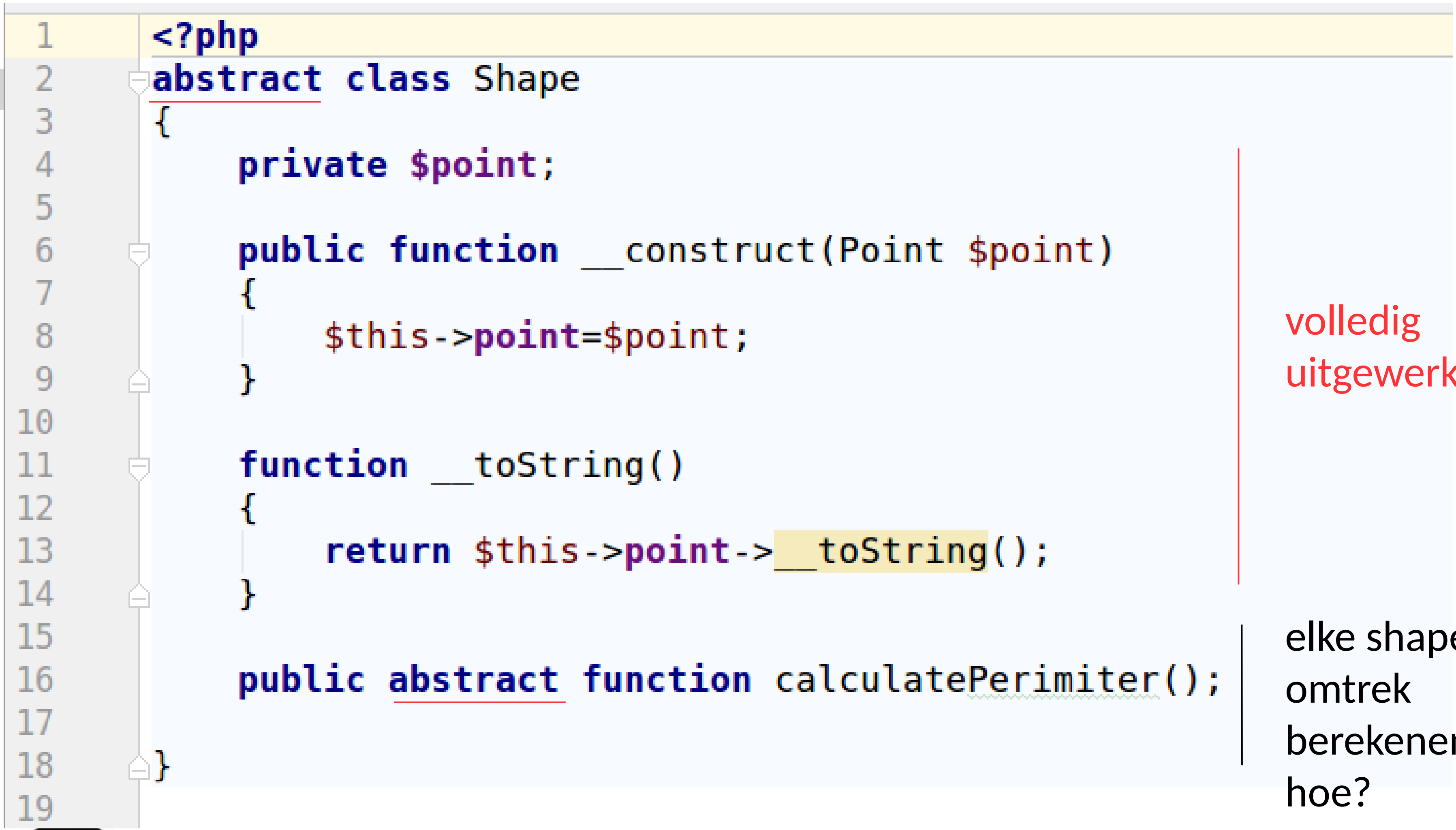


```php
<?php
require_once 'Point.php';
$point1=new Point(1,2);
$point2=new Point(5,11);
print(Point::getCountInitialisations()."\n");
unset($point2);
print(Point::getCountInitialisations());
```

```
/usr/bin/php /home/jan/PhpstormProjects/les_1/app.php

2
1
Process finished with exit code 0
```

static method: ClassName::

Abstract class

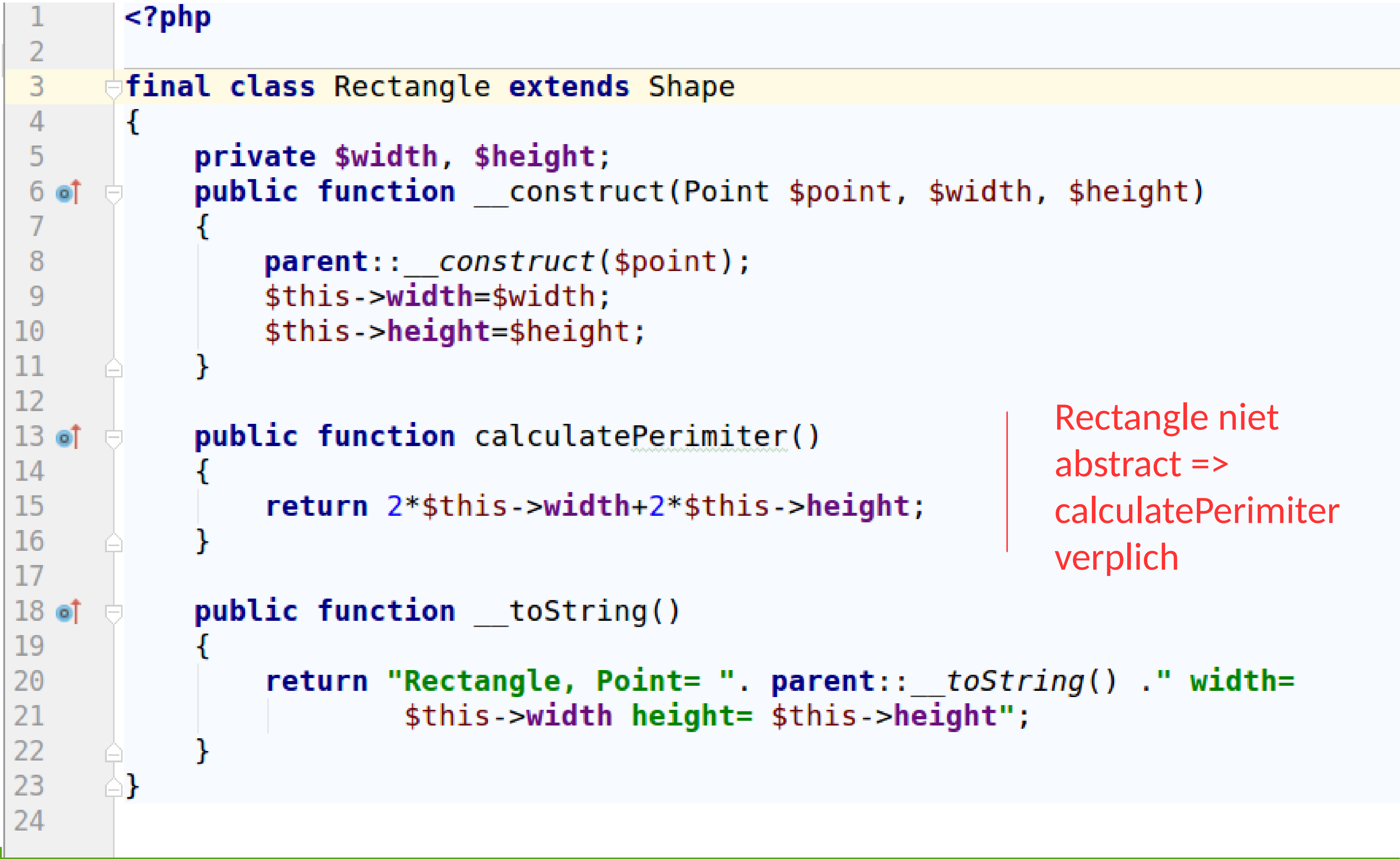


```php
<?php
abstract class Shape
{
    private $point;

    public function __construct(Point $point)
    {
        $this->point=$point;
    }

    function __toString()
    {
        return $this->point->__toString();
    }

    public abstract function calculatePerimiter();

}
```

volledig uitgewerkt

elke shape omtrek berekenen hoe?

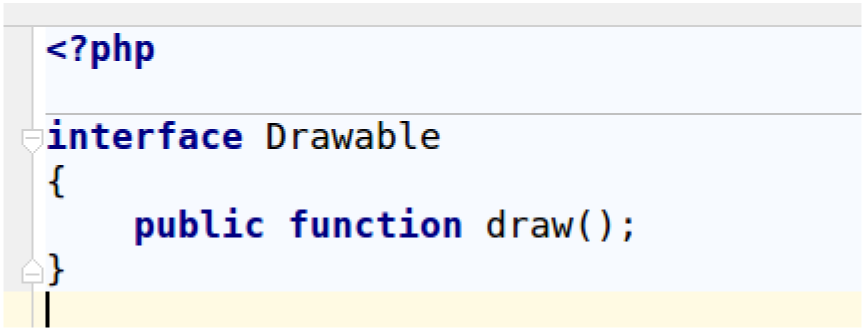
Abstract class



```php
<?php

final class Rectangle extends Shape
{
    private $width, $height;
    public function __construct(Point $point, $width, $height)
    {
        parent::__construct($point);
        $this->width=$width;
        $this->height=$height;
    }

    public function calculatePerimiter()
    {
        return 2*$this->width+2*$this->height;
    }

    public function __toString()
    {
        return "Rectangle, Point= ". parent::__toString() ." width=
                $this->width height= $this->height";
    }
}
```

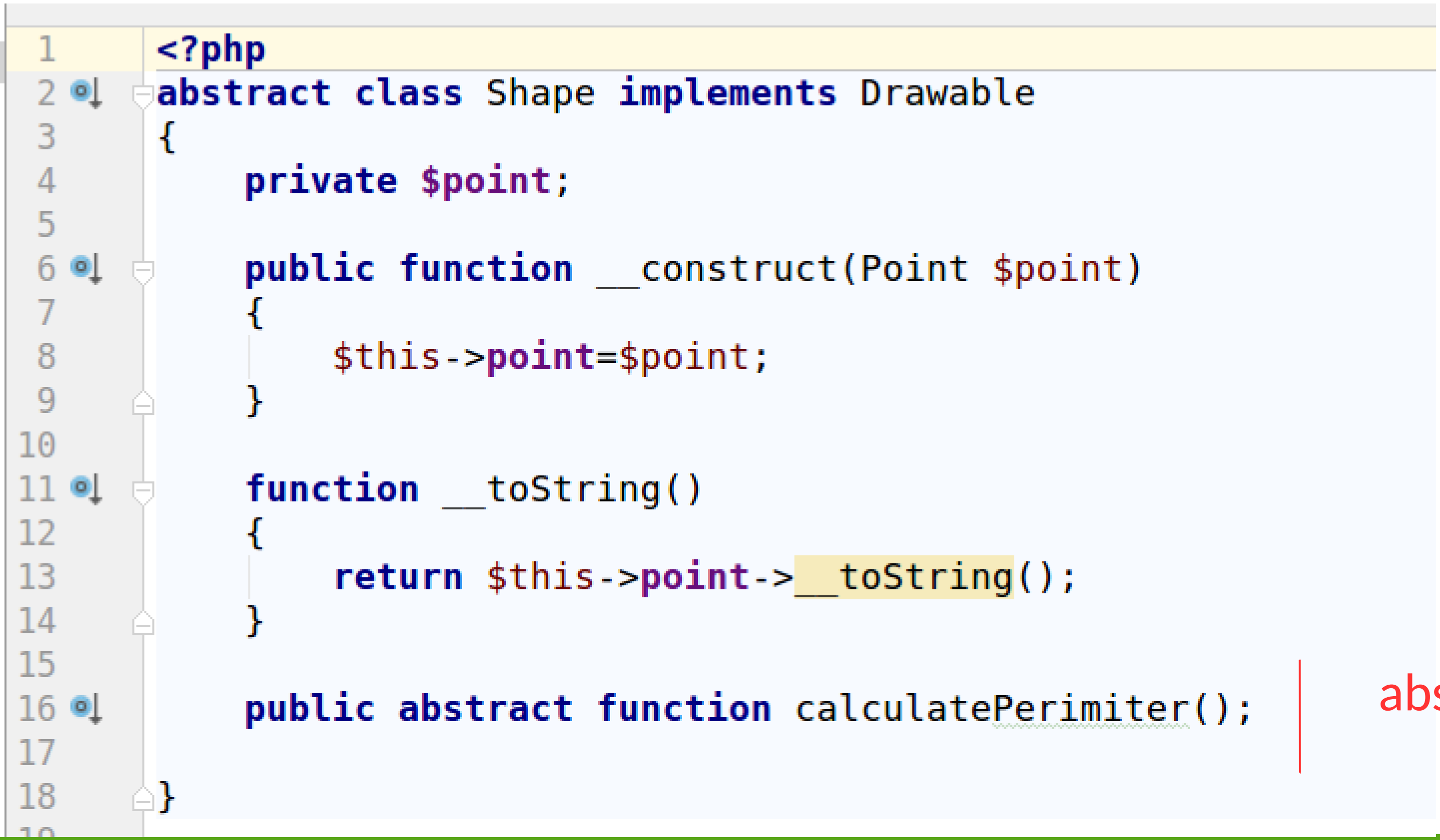
Rectangle niet abstract => calculatePerimiter verplich

Interface



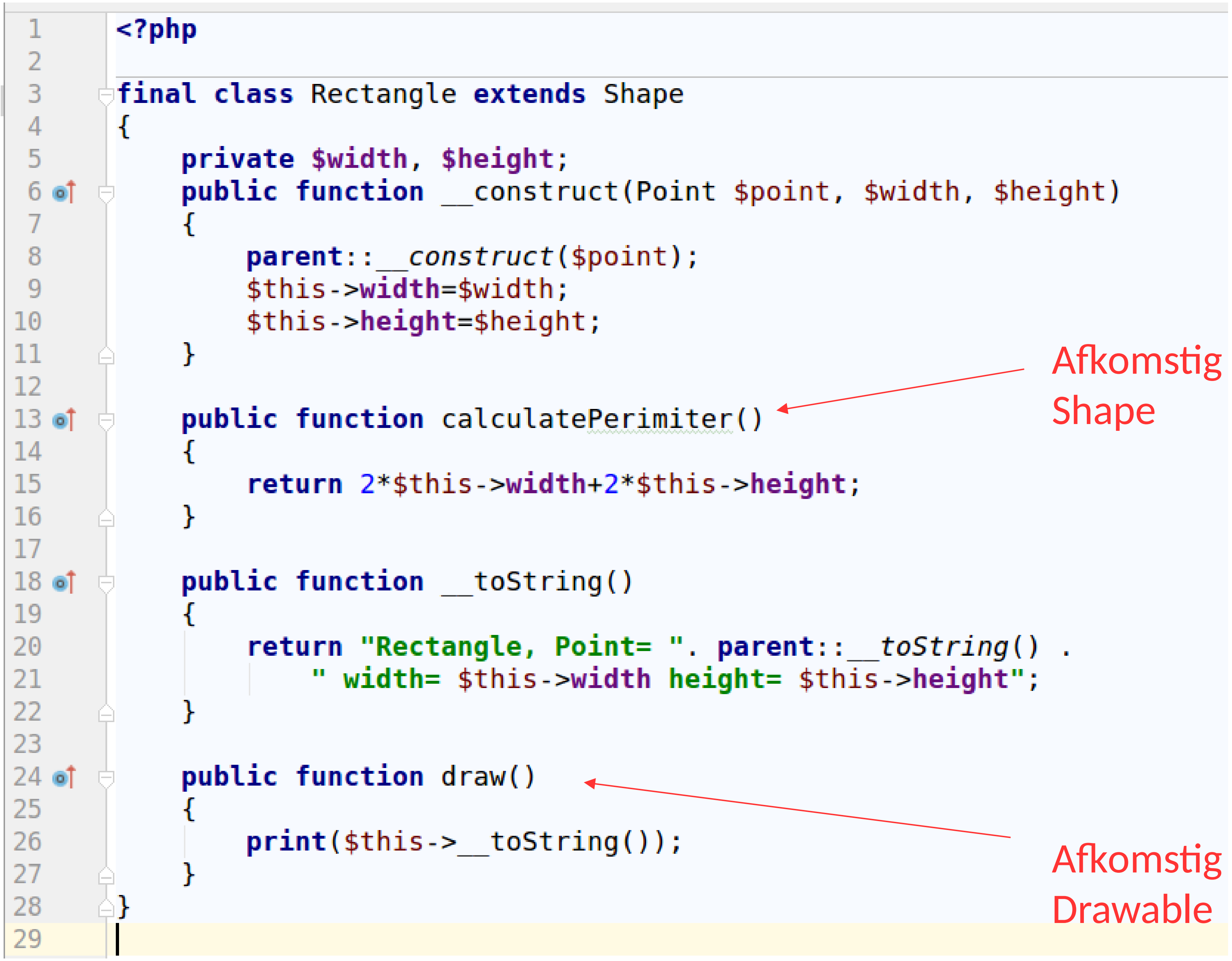
```php
<?php

interface Drawable
{
    public function draw();
}
```

abstracte methode draw



```php
<?php
abstract class Shape implements Drawable
{
    private $point;

    public function __construct(Point $point)
    {
        $this->point=$point;
    }

    function __toString()
    {
        return $this->point->__toString();
    }

    public abstract function calculatePerimiter();

}
```

abstracte methode

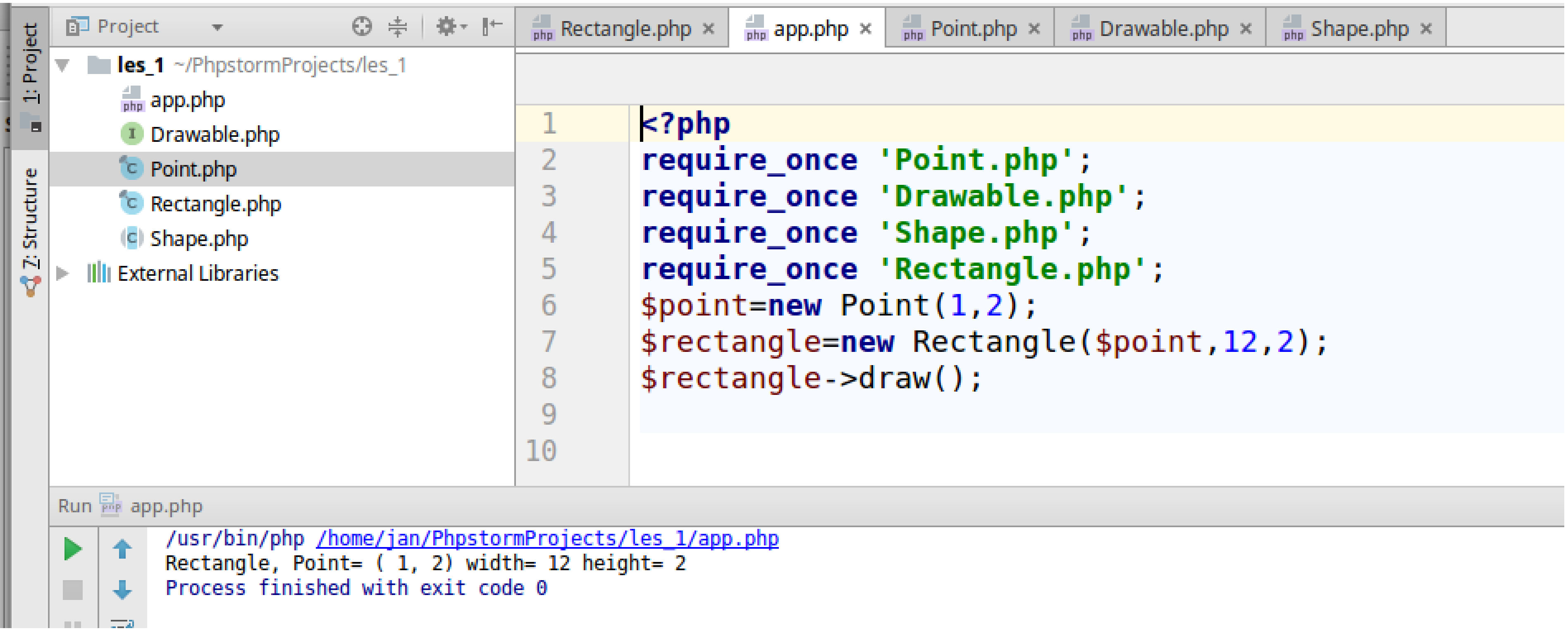


```php
<?php

final class Rectangle extends Shape
{
    private $width, $height;
    public function __construct(Point $point, $width, $height)
    {
        parent::__construct($point);
        $this->width=$width;
        $this->height=$height;
    }

    public function calculatePerimiter()
    {
        return 2*$this->width+2*$this->height;
    }

    public function __toString()
    {
        return "Rectangle, Point= ". parent::__toString() .
            " width= $this->width height= $this->height";
    }

    public function draw()
    {
        print($this->__toString());
    }
}
```

Afkomstig v. Shape

Afkomstig v. Drawable

Interface



```php
<?php
require_once 'Point.php';
require_once 'Drawable.php';
require_once 'Shape.php';
require_once 'Rectangle.php';
$point=new Point(1,2);
$rectangle=new Rectangle($point,12,2);
$rectangle->draw();
```

```
/usr/bin/php /home/jan/PhpstormProjects/les_1/app.php
Rectangle, Point= ( 1, 2) width= 12 height= 2
Process finished with exit code 0
```

Namespaces

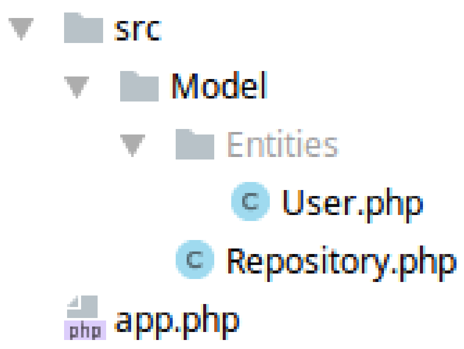
~ Java packages

Nut:

- naming collisions vermijden
- unieke identificatie van klassen / interfaces

Conventie:

- namespaces komen overeen met directory-structuur



```
src
    Model
        Entities
            User.php
        Repository.php
    app.php
```

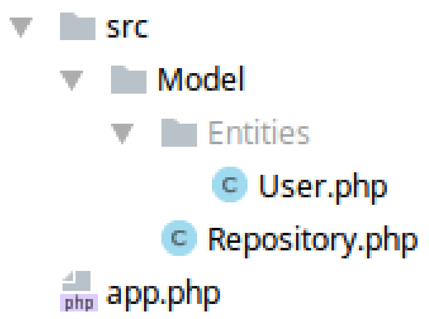
src/Model/Repository.php

```
<?php namespace Model;

class Repository
{
    ...
}
```

src/Model/Entities/User.php

```
<?php namespace Model\Entities;

class User
{
    ...
}
```

Namespaces



In het bestand app.php wordt geen namespace gedefinieerd. Alle code in dit bestand hoort in de default-namespace (\ ).

app.php

```
<?php
require_once 'src/Model/Repository.php';
require_once 'src/Model/Entities/User.php';
$user = new Model\Entities\User();
$repository = new Model\Repository();
```

Namespaces

use: verkorte notatie

app.php

```
<?php
require_once 'src/Model/Repository.php';
require_once 'src/Model/Entities/User.php';
$user = new Model\Entities\User();
$repository = new Model\Repository();
```

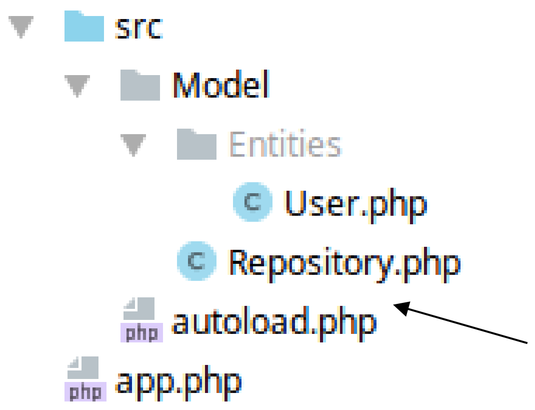
$$\downarrow$$

app.php

```
<?php
require_once 'src/Model/Repository.php';
require_once 'src/Model/Entities/User.php';
use Model\Repository;
use Model\Entities\User;
$user = new User();
$repository = new Repository();
```

Autoloading

Een autoloader (srr/autoload.php) vereenvoudigt het ophalen van code:



```
src
    Model
        Entities
            User.php
        Repository.php
    autoload.php
app.php
```

app.php

```
<?php
require_once 'src/autoload.php';
use Model\Repository;
use Model\Entities\User;
$user = new User();
$repository = new Repository();
```

Autoloading

src/autoload.php

```
<?php
if (!function_exists('classAutoLoader')) {
    function classAutoLoader($className)
    {
        $fileName = 'src/'.
                    str_replace('\\', '/', $className).
                    '.php';
        if(file_exists($fileName)) {
            require_once $fileName;
        }
    }
}
spl_autoload_register('classAutoLoader');
```

Voor elke niet gekende klasse wordt de functie classAutoLoader aangeroepen. In deze functie wordt

Model\Repository	omgezet naar	src/Model/Repository.php
Model\Entities	omgezet naar	src/Model/Entities/User.php

Autoloading: composer

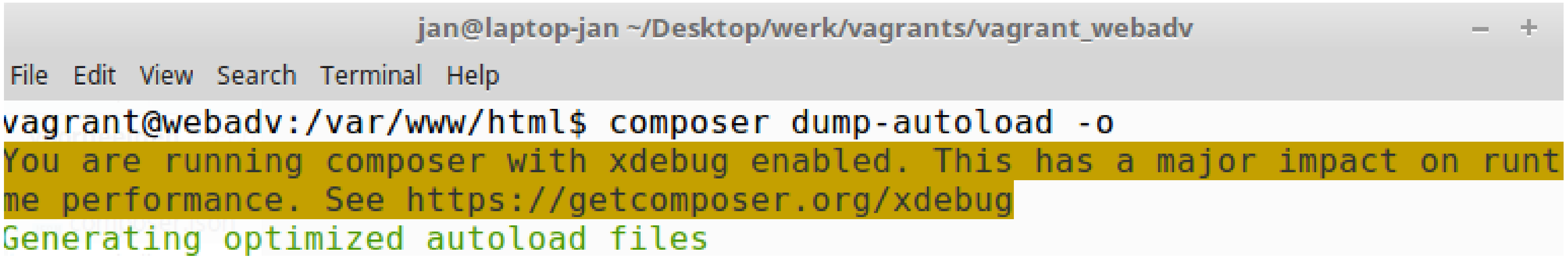
Dependency Manager for PHP

Dependencies (bv Monolog) downloaden van repository

Maakt ook autoloader voor afgehaalde code (en eigen code).

composer.json

```
{
    "autoload": {
        "psr-4": {
            "Model\\": "src/Model/",
            "Model\\Entities\\": "src/Model/Entities"
        }
    }
}
```


jan@laptop-jan ~/Desktop/werk/vagrants/vagrant_webadv
File Edit View Search Terminal Help

```
vagrant@webadv:/var/www/html$ composer dump-autoload -o
You are running composer with xdebug enabled. This has a major impact on runt
me performance. See https://getcomposer.org/xdebug
Generating optimized autoload files
```

app.php

```
<?php
require_once 'vendor/autoload.php';
use Model\Repository;
use Model\Entities\User;
$user = new User();
$repository = new Repository();
```

https://getcomposer.org/doc/01-basic-usage.md#autoloading