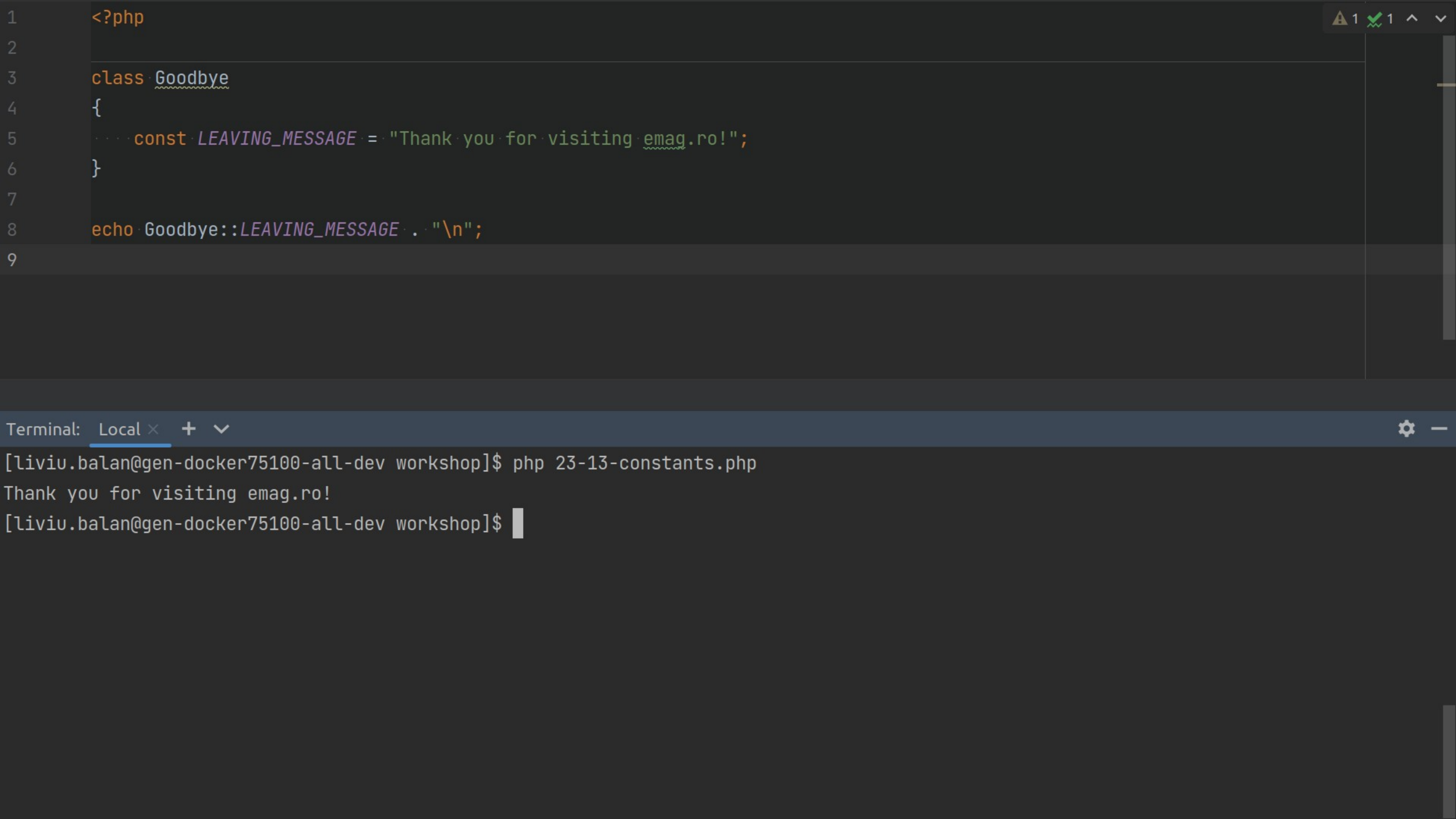
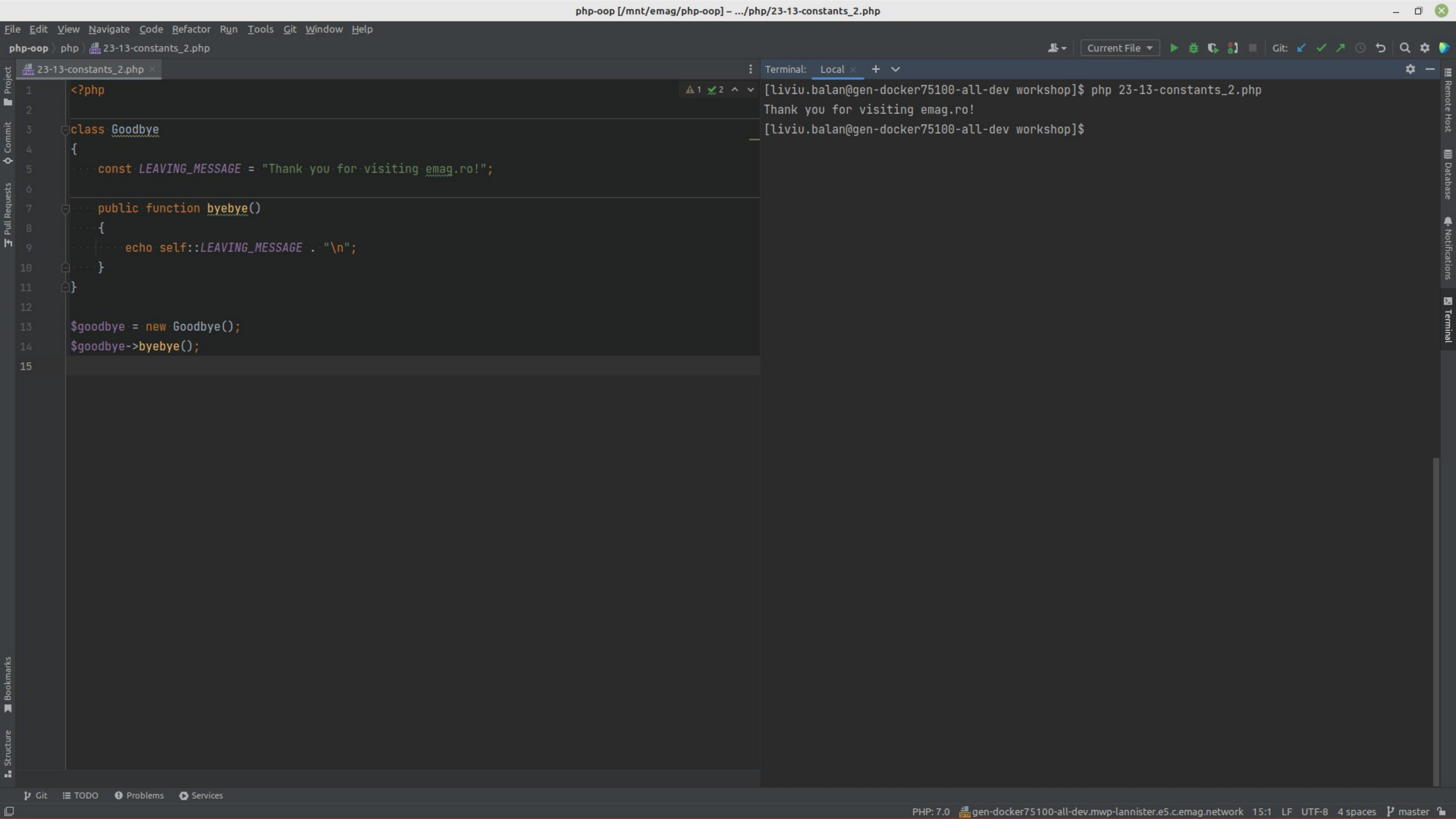
Workshop PHP
Part 3

Liviu Bălan



```php
<?php

class Goodbye
{
    const LEAVING_MESSAGE = "Thank you for visiting emag.ro!";
}

echo Goodbye::LEAVING_MESSAGE . "\n";
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-13-constants.php
Thank you for visiting emag.ro!
[liviu.balan@gen-docker75100-all-dev workshop]$
```



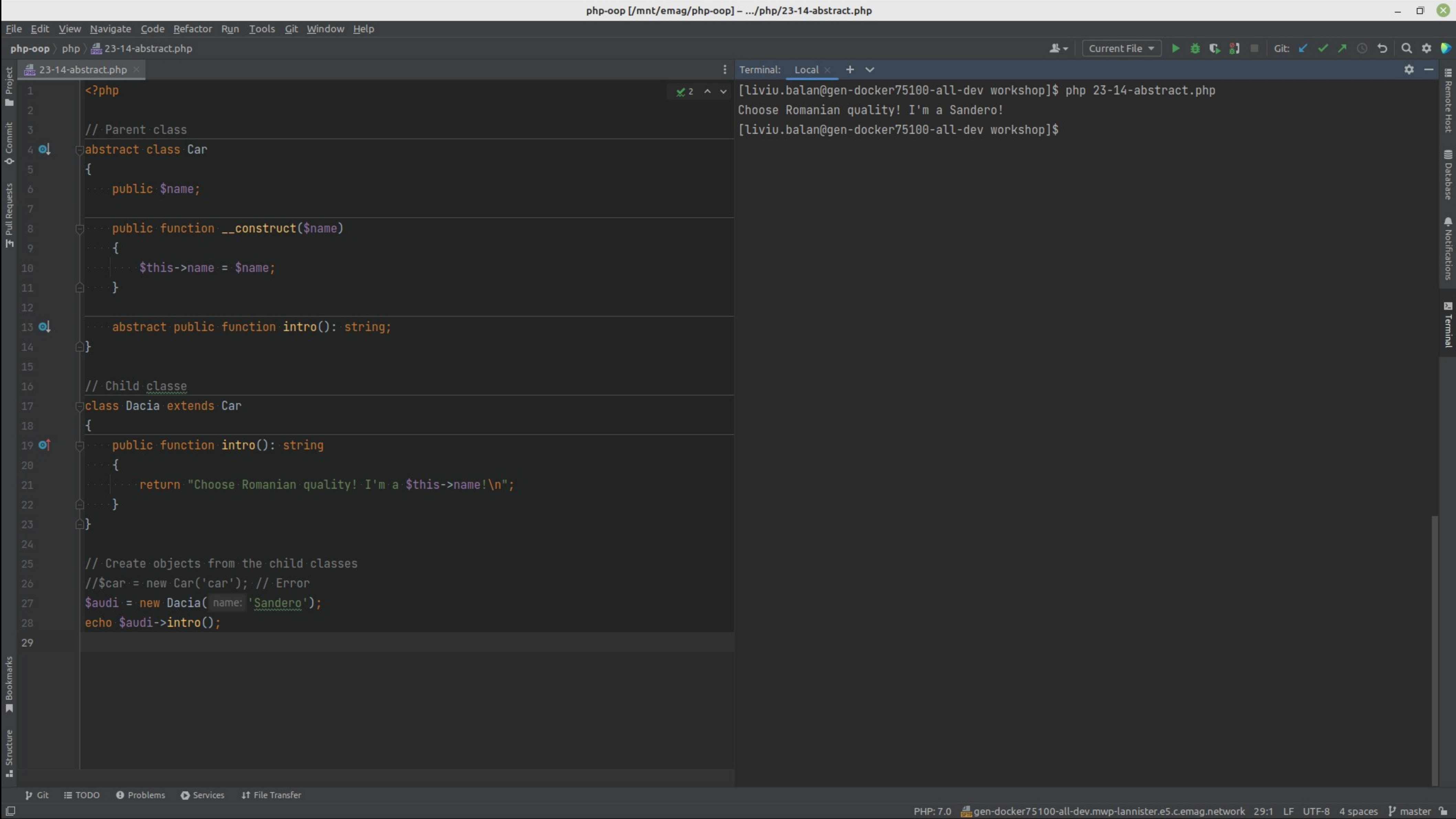
php-oop [/mnt/emag/php-oop] – .../php/23-13-constants_2.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop 〉 php 〉 23-13-constants_2.php
Current File
Terminal: Local
Git TODO Problems Services
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 15:1 LF UTF-8 4 spaces master

```php
<?php

class Goodbye
{
    const LEAVING_MESSAGE = "Thank you for visiting emag.ro!";

    public function byebye()
    {
        echo self::LEAVING_MESSAGE . "\n";
    }
}

$goodbye = new Goodbye();
$goodbye->byebye();

```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-13-constants_2.php
Thank you for visiting emag.ro!
[liviu.balan@gen-docker75100-all-dev workshop]$
```

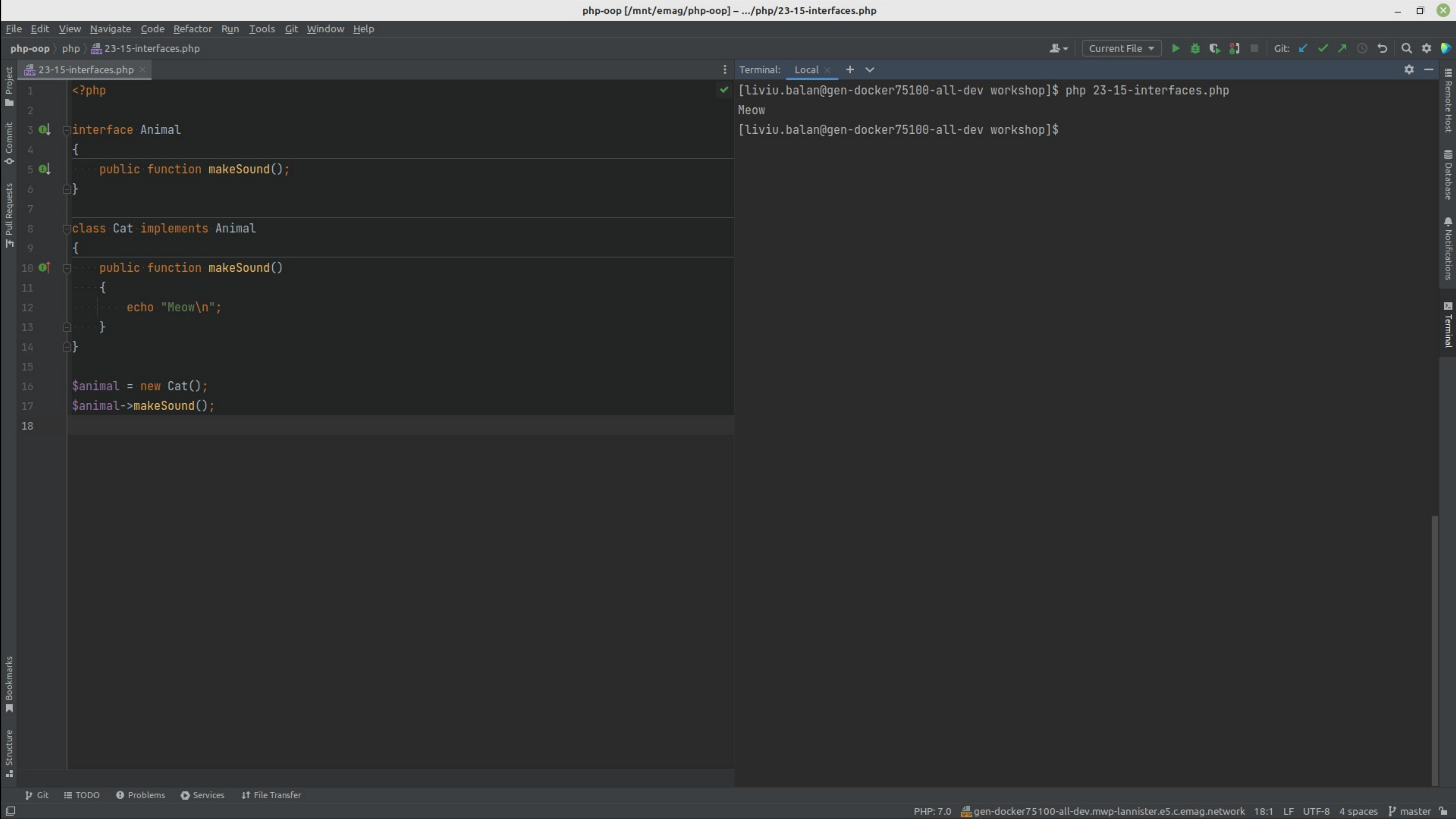
Interfaces vs. Abstract Classes

- Interfaces cannot have properties, while abstract classes can
- All interface methods must be public, while abstract class methods is public or protected
- All methods in an interface are abstract, so they cannot be implemented in code and the abstract keyword is not necessary
- Classes can implement an interface while inheriting from another class at the same time


php-oop [/mnt/emag/php-oop] – .../php/23-14-abstract.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop 〉 php 〉 23-14-abstract.php
Current File
23-14-abstract.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 29:1 LF UTF-8 4 spaces master

```php
<?php

// Parent class
abstract class Car
{
    public $name;

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

    abstract public function intro(): string;
}

// Child classe
class Dacia extends Car
{
    public function intro(): string
    {
        return "Choose Romanian quality! I'm a $this->name!\n";
    }
}

// Create objects from the child classes
//$car = new Car('car'); // Error
$audi = new Dacia( name: 'Sandero');
echo $audi->intro();
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-14-abstract.php
Choose Romanian quality! I'm a Sandero!
[liviu.balan@gen-docker75100-all-dev workshop]$
```

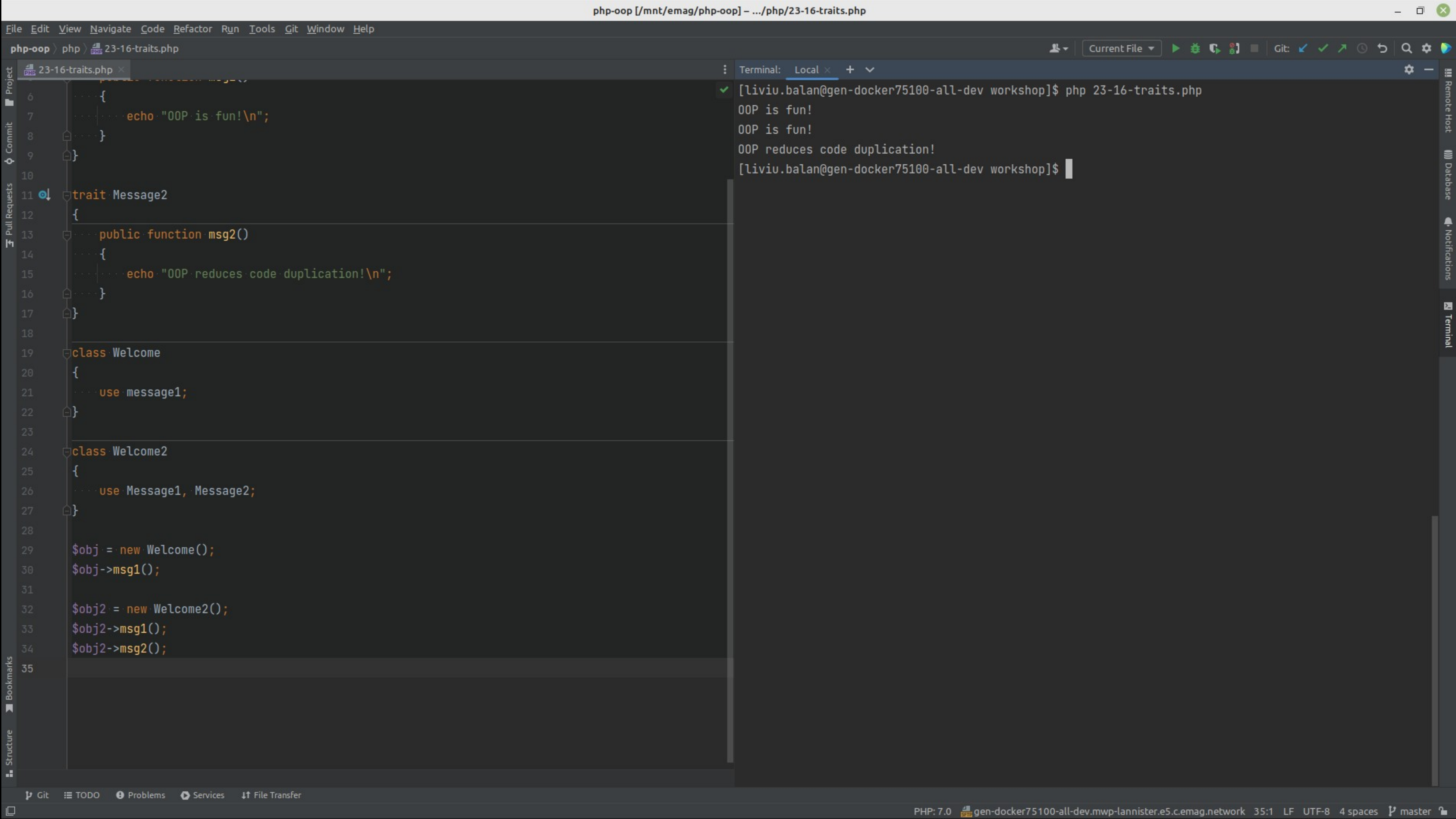

php-oop [/mnt/emag/php-oop] – .../php/23-15-interfaces.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop 〉 php 〉 23-15-interfaces.php
Current File
Git:
23-15-interfaces.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 18:1 LF UTF-8 4 spaces master

```php
<?php

interface Animal
{
    public function makeSound();
}

class Cat implements Animal
{
    public function makeSound()
    {
        echo "Meow\n";
    }
}

$animal = new Cat();
$animal->makeSound();
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-15-interfaces.php
Meow
[liviu.balan@gen-docker75100-all-dev workshop]$
```

Traits

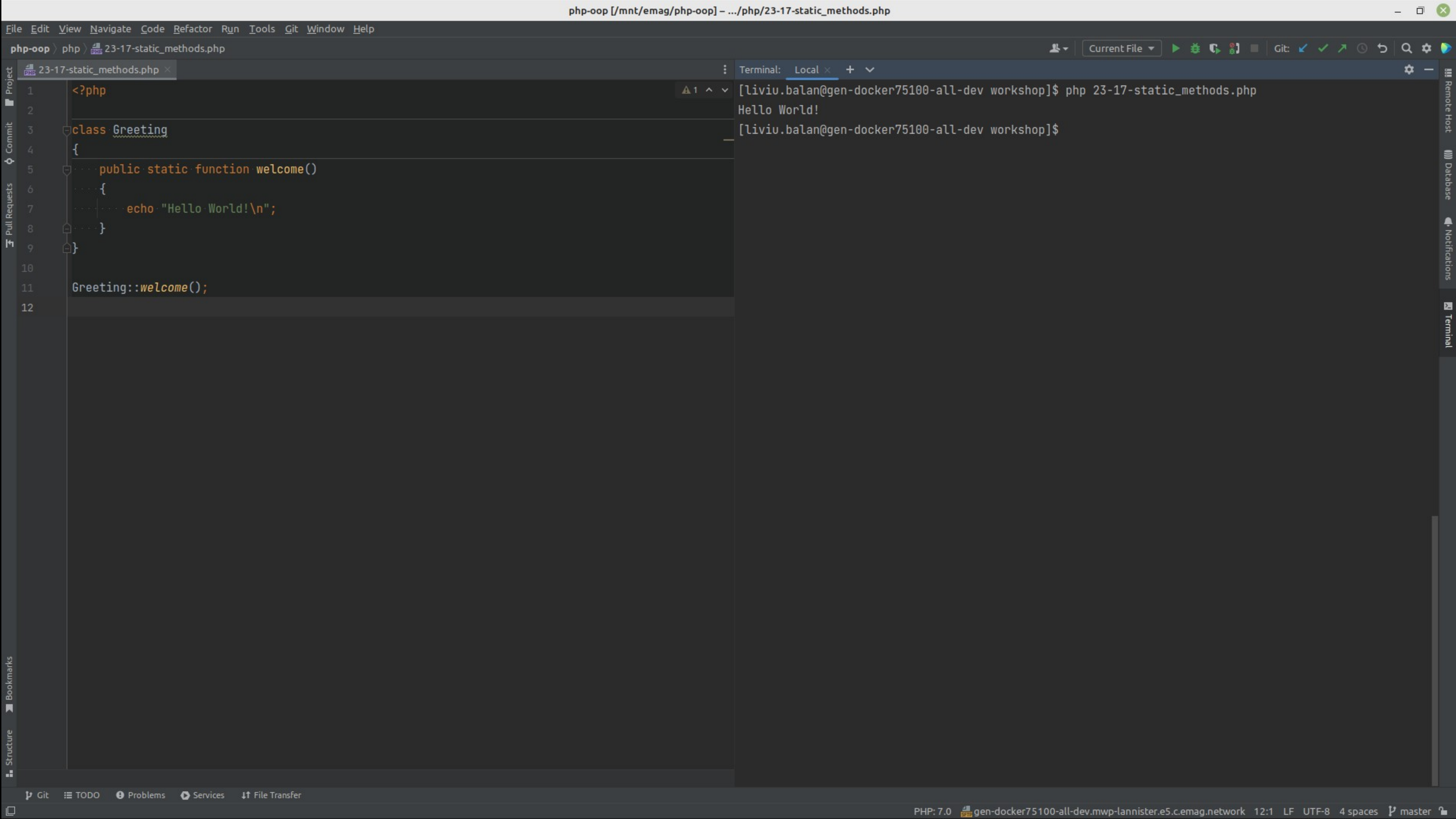
- PHP only supports single inheritance: a child class can inherit only from one single parent.
- What if a class needs to inherit multiple behaviors? OOP traits solve this problem.
- Traits are used to declare methods that can be used in multiple classes.
- Traits can have methods and abstract methods that can be used in multiple classes, and the methods can have any access modifier (public, private, or protected).


php-oop [/mnt/emag/php-oop] – .../php/23-16-traits.php

```php
    {
        echo "OOP is fun!\n";
    }
}

trait Message2
{
    public function msg2()
    {
        echo "OOP reduces code duplication!\n";
    }
}

class Welcome
{
    use message1;
}

class Welcome2
{
    use Message1, Message2;
}

$obj = new Welcome();
$obj->msg1();

$obj2 = new Welcome2();
$obj2->msg1();
$obj2->msg2();
```

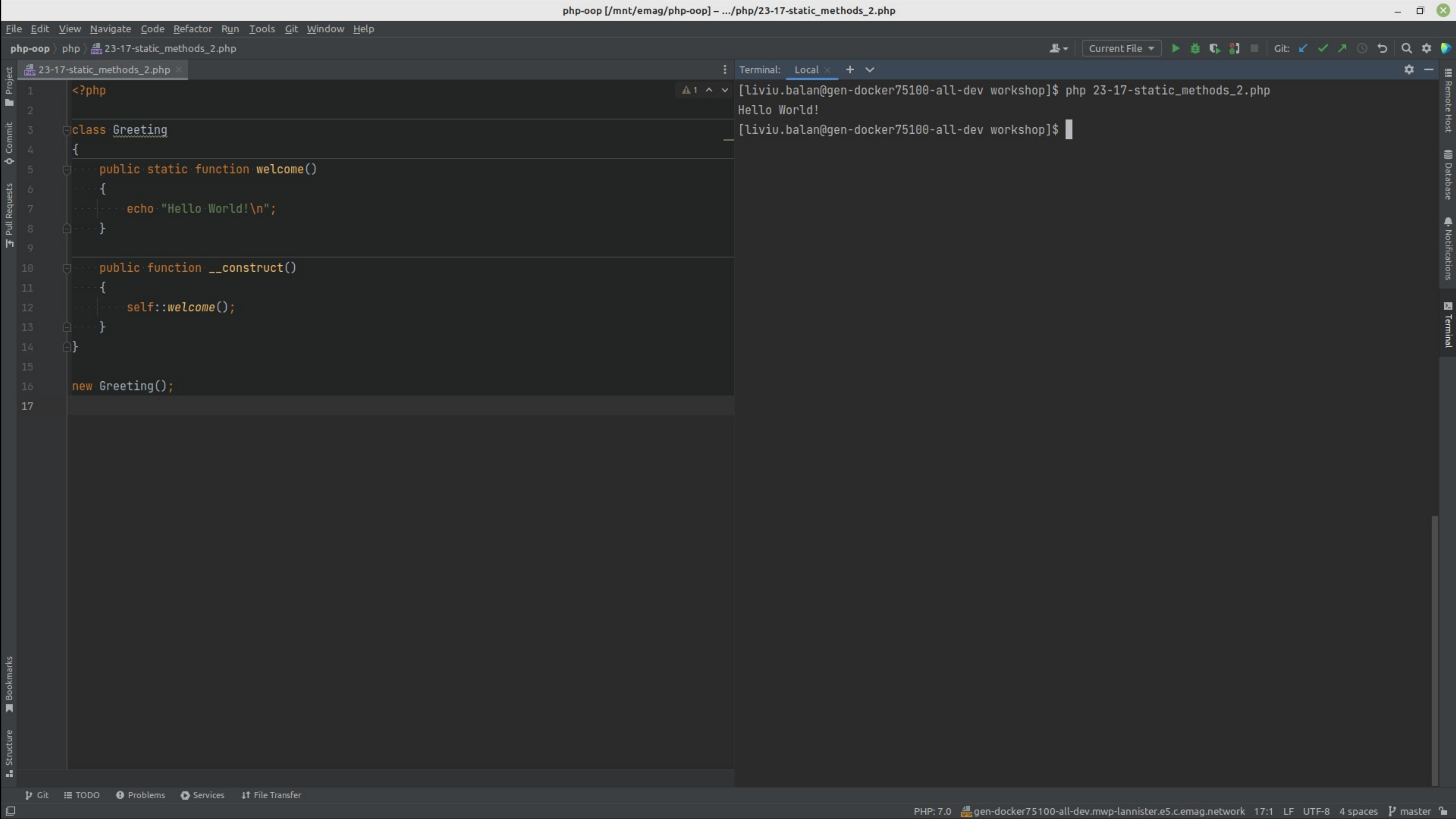
```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-16-traits.php
OOP is fun!
OOP is fun!
OOP reduces code duplication!
[liviu.balan@gen-docker75100-all-dev workshop]$
```

PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 35:1 LF UTF-8 4 spaces master


php-oop [/mnt/emag/php-oop] – .../php/23-17-static_methods.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop 〉 php 〉 23-17-static_methods.php
Current File
23-17-static_methods.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 12:1 LF UTF-8 4 spaces master

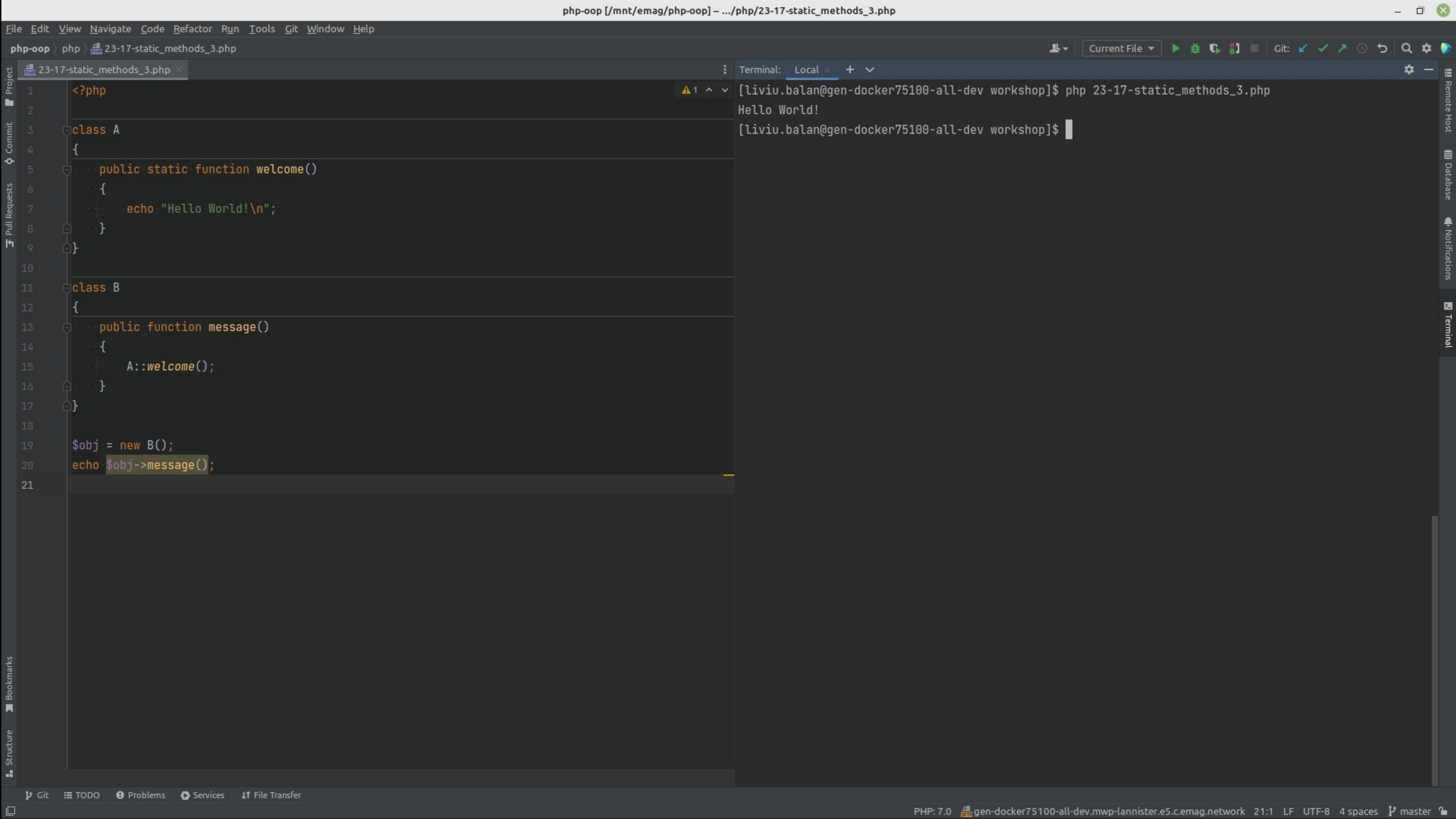
```php
<?php

class Greeting
{
    public static function welcome()
    {
        echo "Hello World!\n";
    }
}

Greeting::welcome();
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-17-static_methods.php
Hello World!
[liviu.balan@gen-docker75100-all-dev workshop]$
```



```php
<?php

class Greeting
{
    public static function welcome()
    {
        echo "Hello World!\n";
    }

    public function __construct()
    {
        self::welcome();
    }
}

new Greeting();
```

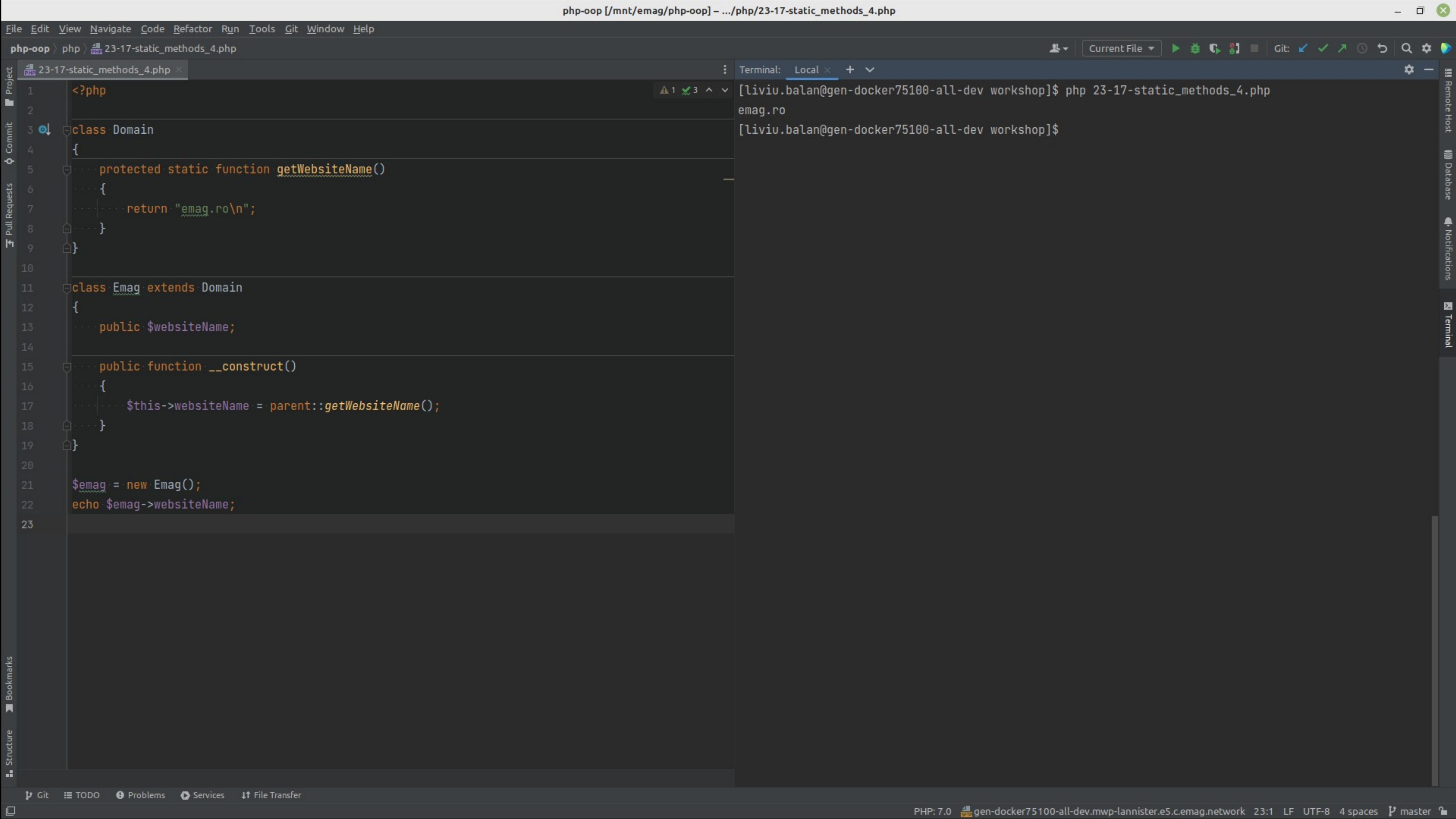
```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-17-static_methods_2.php
Hello World!
[liviu.balan@gen-docker75100-all-dev workshop]$
```


php-oop [/mnt/emag/php-oop] – .../php/23-17-static_methods_3.php

```php
<?php

class A
{
    public static function welcome()
    {
        echo "Hello World!\n";
    }
}

class B
{
    public function message()
    {
        A::welcome();
    }
}

$obj = new B();
echo $obj->message();
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-17-static_methods_3.php
Hello World!
[liviu.balan@gen-docker75100-all-dev workshop]$
```

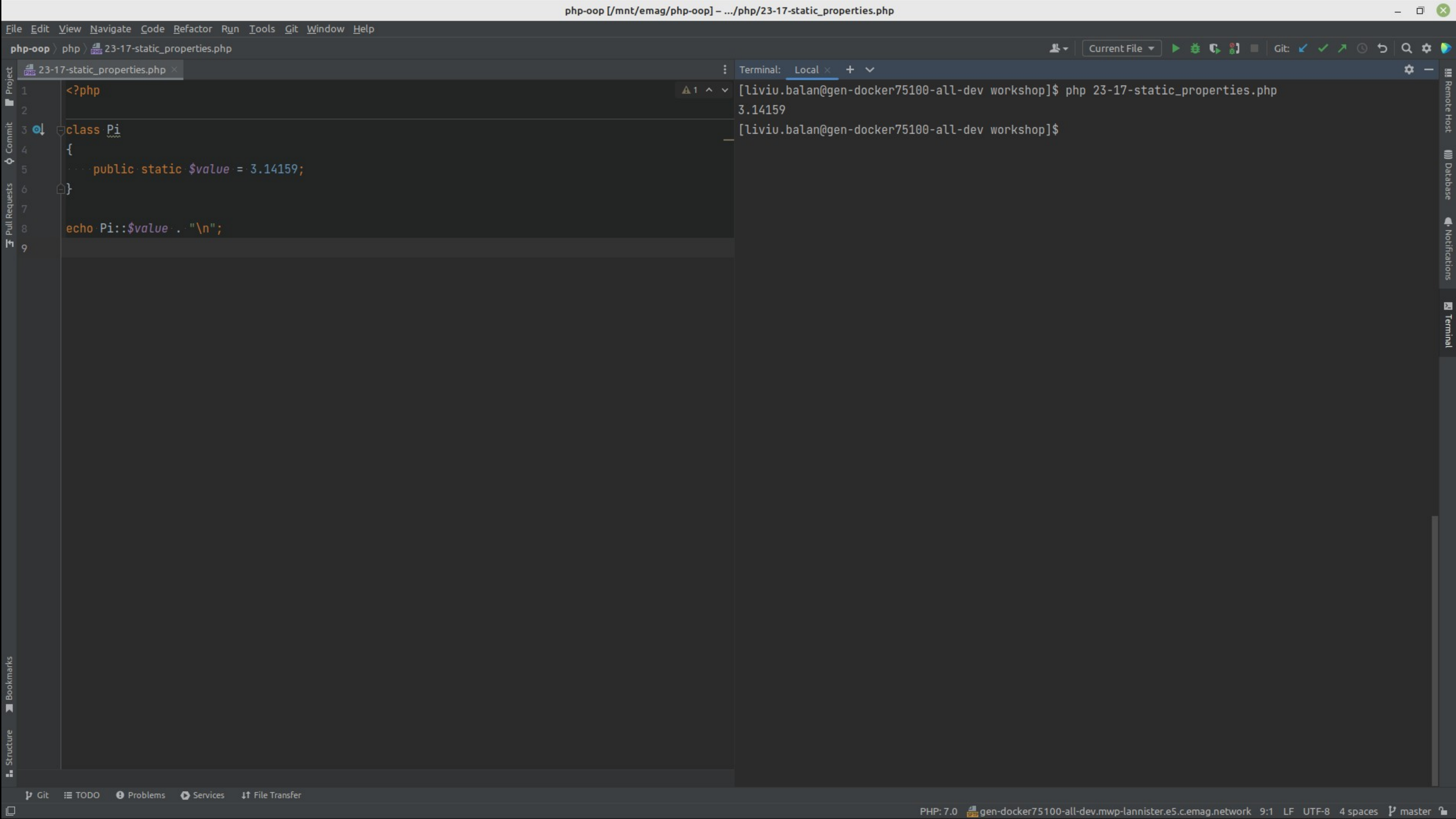
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop › php › 23-17-static_methods_3.php
Current File
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 21:1 LF UTF-8 4 spaces master


php-oop [/mnt/emag/php-oop] – .../php/23-17-static_methods_4.php

```php
<?php

class Domain
{
    protected static function getWebsiteName()
    {
        return "emag.ro\n";
    }
}

class Emag extends Domain
{
    public $websiteName;

    public function __construct()
    {
        $this->websiteName = parent::getWebsiteName();
    }
}

$emag = new Emag();
echo $emag->websiteName;

```

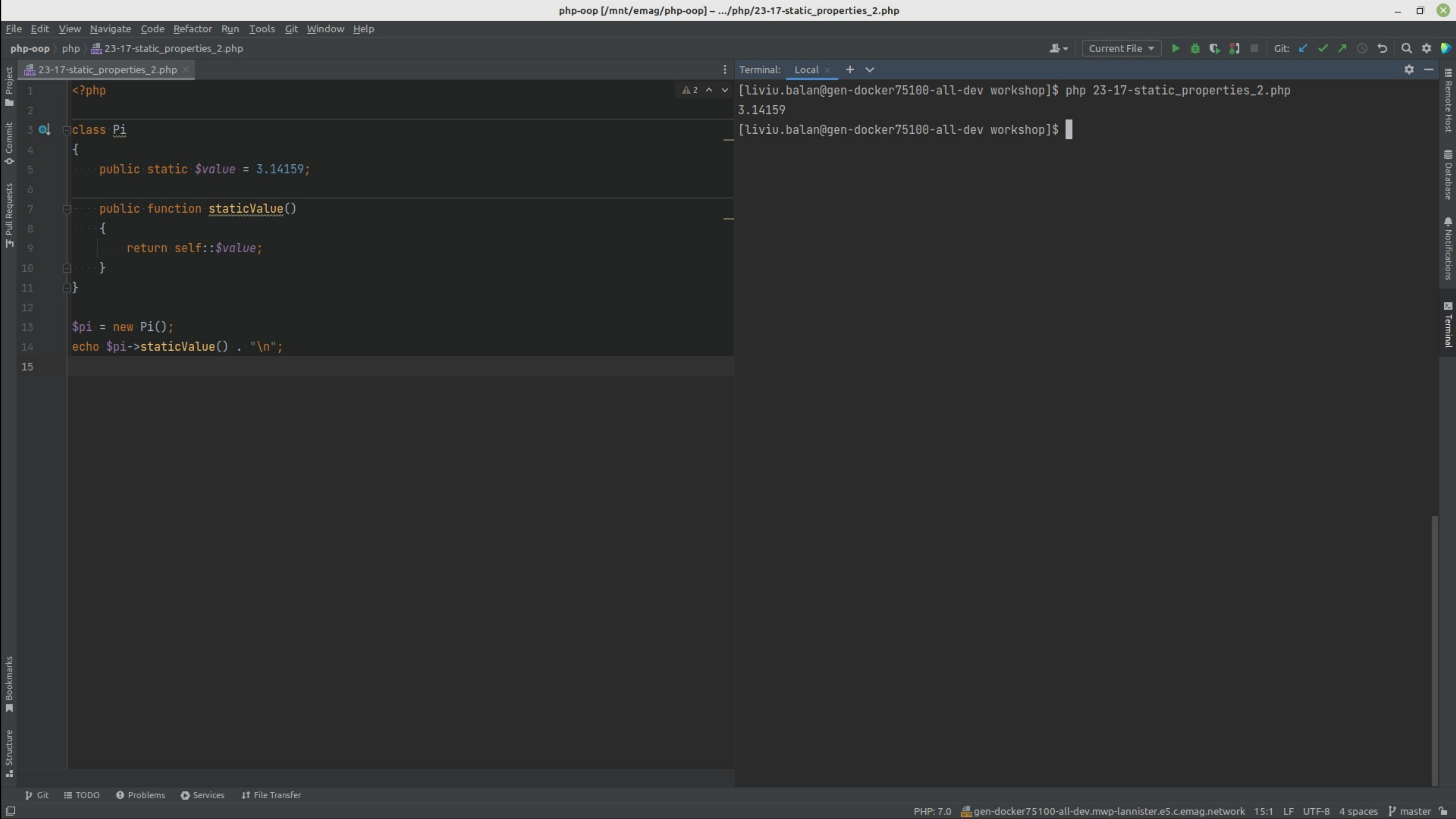
```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-17-static_methods_4.php
emag.ro
[liviu.balan@gen-docker75100-all-dev workshop]$
```

File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop › php › 23-17-static_methods_4.php
Current File
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 23:1 LF UTF-8 4 spaces master


php-oop [/mnt/emag/php-oop] – .../php/23-17-static_properties.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop 〉 php 〉 23-17-static_properties.php
Current File
Git:
23-17-static_properties.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 9:1 LF UTF-8 4 spaces master

```php
<?php

class Pi
{
    public static $value = 3.14159;
}

echo Pi::$value . "\n";
```

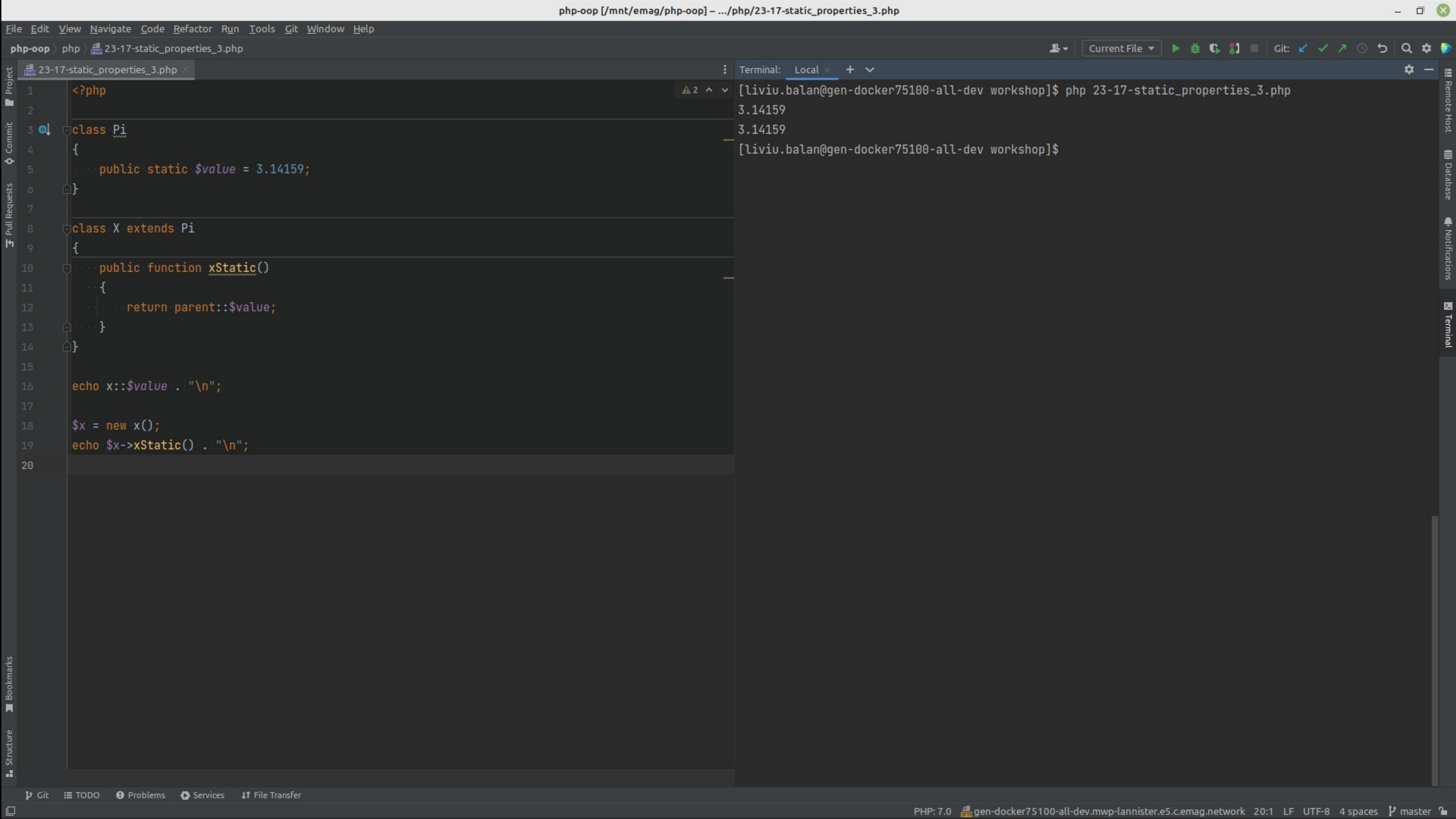
```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-17-static_properties.php
3.14159
[liviu.balan@gen-docker75100-all-dev workshop]$
```



php-oop [/mnt/emag/php-oop] – .../php/23-17-static_properties_2.php

```php
<?php

class Pi
{
    public static $value = 3.14159;

    public function staticValue()
    {
        return self::$value;
    }
}

$pi = new Pi();
echo $pi->staticValue() . "\n";

```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-17-static_properties_2.php
3.14159
[liviu.balan@gen-docker75100-all-dev workshop]$
```

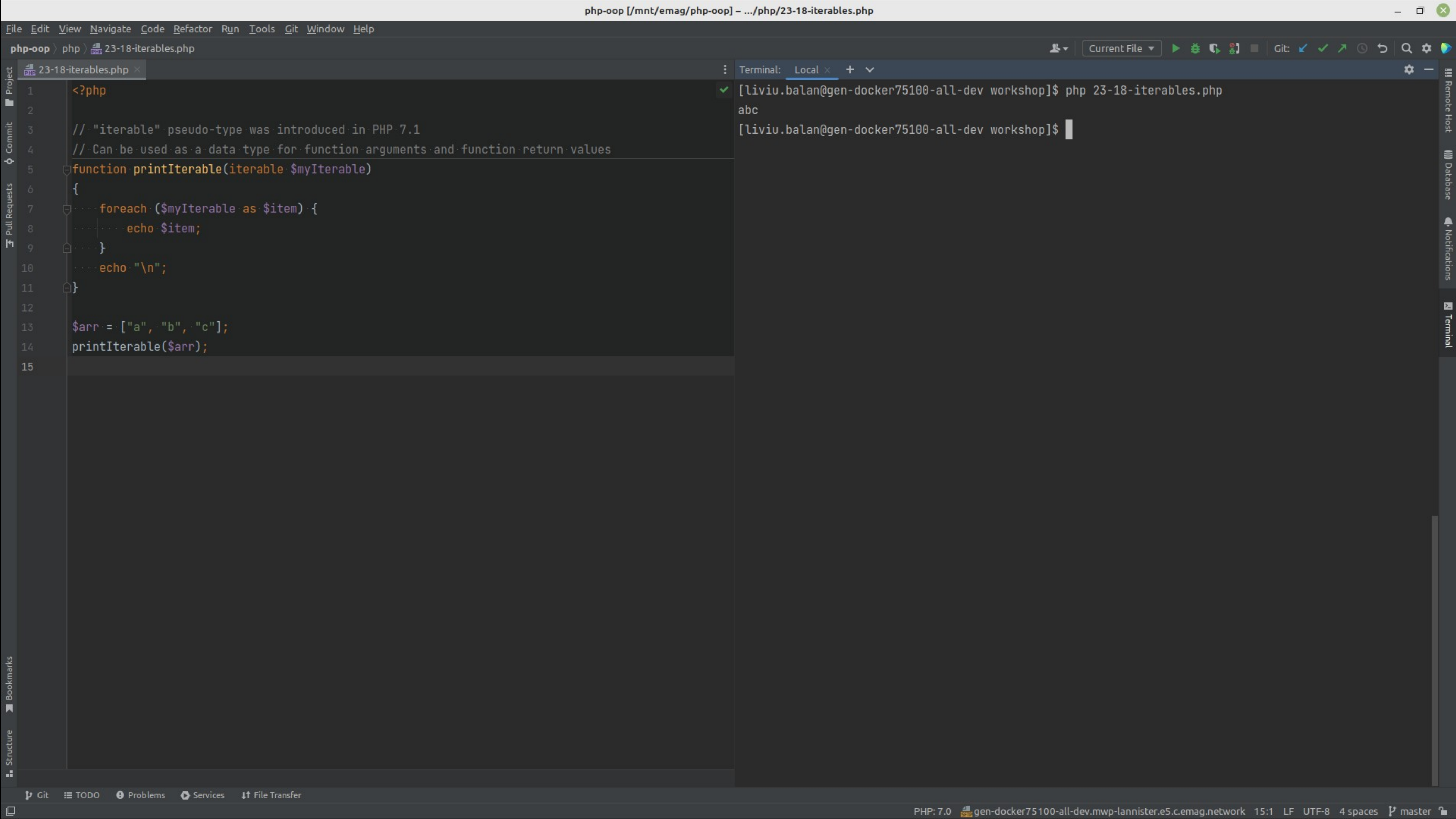


```php
<?php

class Pi
{
    public static $value = 3.14159;
}

class X extends Pi
{
    public function xStatic()
    {
        return parent::$value;
    }
}

echo x::$value . "\n";

$x = new x();
echo $x->xStatic() . "\n";
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-17-static_properties_3.php
3.14159
3.14159
[liviu.balan@gen-docker75100-all-dev workshop]$
```

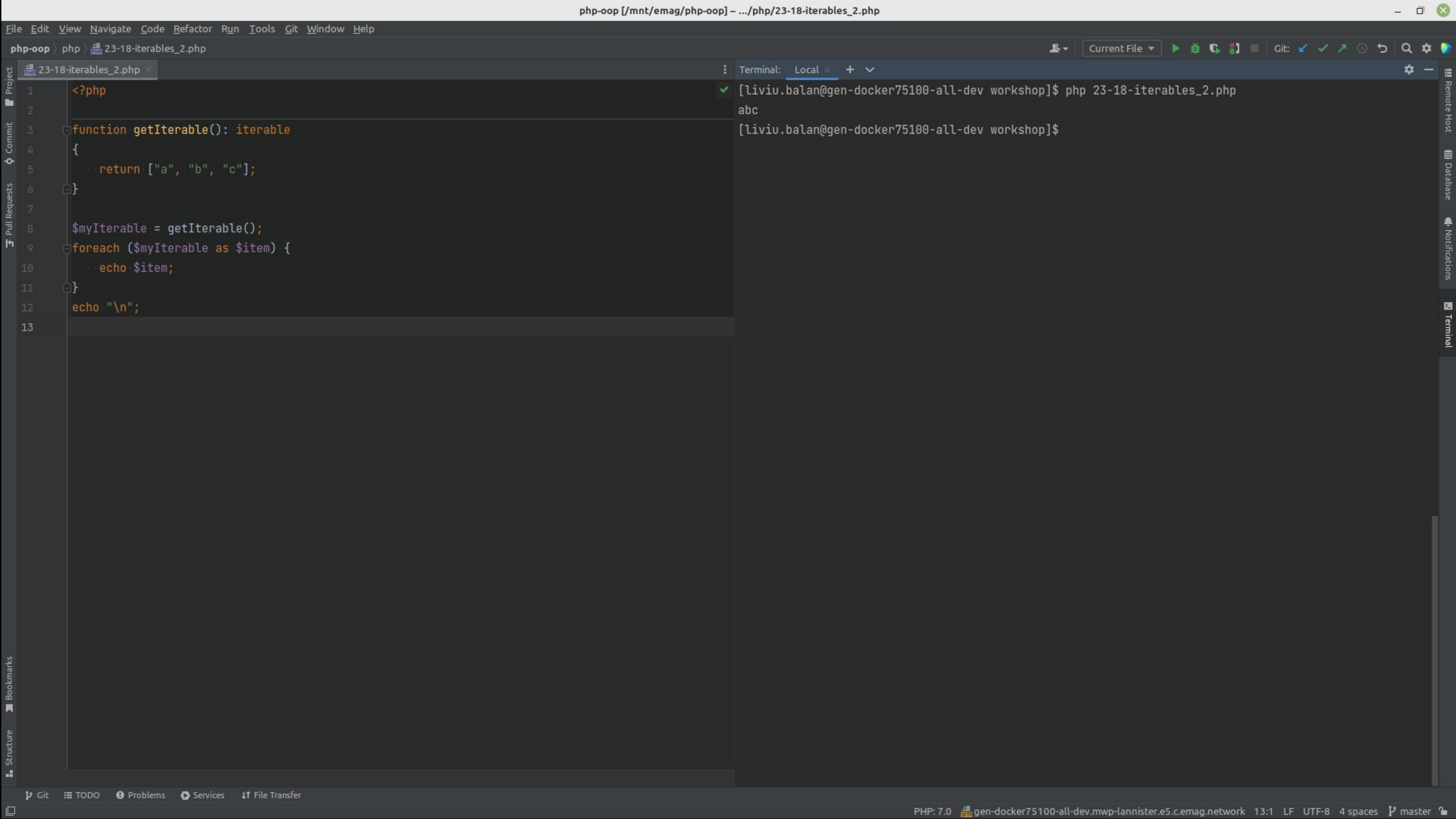
Creating Iterables

- All arrays are iterables
- Any object that implements the **Iterator** interface can be used as an argument of a function that requires an iterable


php-oop [/mnt/emag/php-oop] – .../php/23-18-iterables.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop 〉 php 〉 23-18-iterables.php
Current File
Git:
23-18-iterables.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 15:1 LF UTF-8 4 spaces master

```php
<?php

// "iterable" pseudo-type was introduced in PHP 7.1
// Can be used as a data type for function arguments and function return values
function printIterable(iterable $myIterable)
{
    foreach ($myIterable as $item) {
        echo $item;
    }
    echo "\n";
}

$arr = ["a", "b", "c"];
printIterable($arr);
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-18-iterables.php
abc
[liviu.balan@gen-docker75100-all-dev workshop]$
```



php-oop [/mnt/emag/php-oop] – .../php/23-18-iterables_2.php

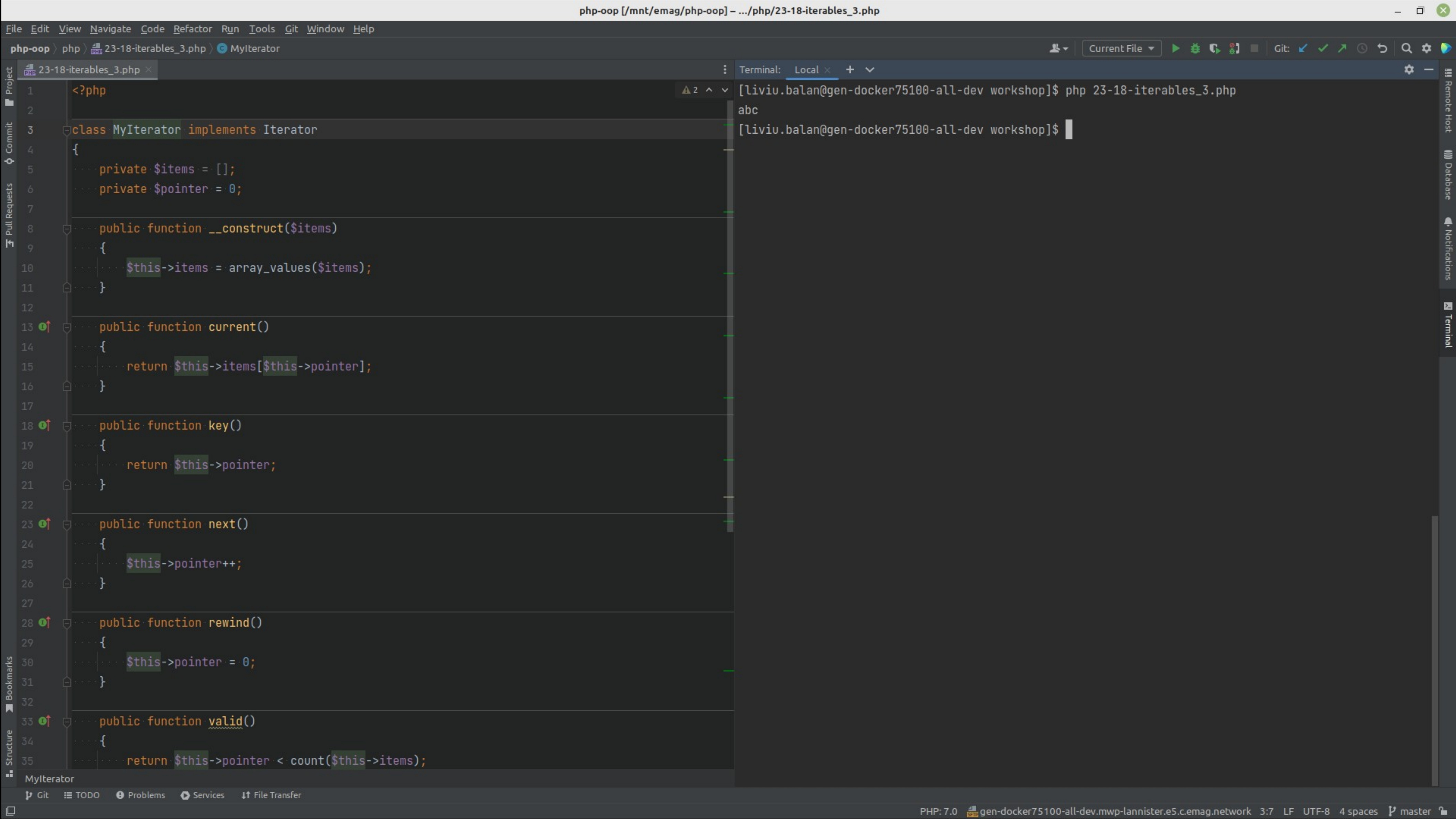
```php
<?php

function getIterable(): iterable
{
    return ["a", "b", "c"];
}

$myIterable = getIterable();
foreach ($myIterable as $item) {
    echo $item;
}
echo "\n";
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-18-iterables_2.php
abc
[liviu.balan@gen-docker75100-all-dev workshop]$
```

File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop › php › 23-18-iterables_2.php
Current File
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 13:1 LF UTF-8 4 spaces master

Iterator methods

- **current()** - Returns the element that the pointer is currently pointing to. It can be any data type
- **key()** Returns the key associated with the current element in the list. It can only be an integer, float, boolean or string
- **next()** Moves the pointer to the next element in the list
- **rewind()** Moves the pointer to the first element in the list
- **valid()** If the internal pointer is not pointing to any element (for example, if next() was called at the end of the list), this should return false. It returns true in any other case



```php
<?php

class MyIterator implements Iterator
{
    private $items = [];
    private $pointer = 0;

    public function __construct($items)
    {
        $this->items = array_values($items);
    }

    public function current()
    {
        return $this->items[$this->pointer];
    }

    public function key()
    {
        return $this->pointer;
    }

    public function next()
    {
        $this->pointer++;
    }

    public function rewind()
    {
        $this->pointer = 0;
    }

    public function valid()
    {
        return $this->pointer < count($this->items);
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 23-18-iterables_3.php
abc
[liviu.balan@gen-docker75100-all-dev workshop]$
```

php-oop [/mnt/emag/php-oop] – .../php/23-18-iterables_3.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop › php › 23-18-iterables_3.php › MyIterator
Current File
23-18-iterables_3.php
Terminal: Local
MyIterator
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 3:7 LF UTF-8 4 spaces master

Design patterns

- https://refactoring.guru/design-patterns/php

Design Patterns in PHP

refactoring.guru/design-patterns/php

SPRING SALE

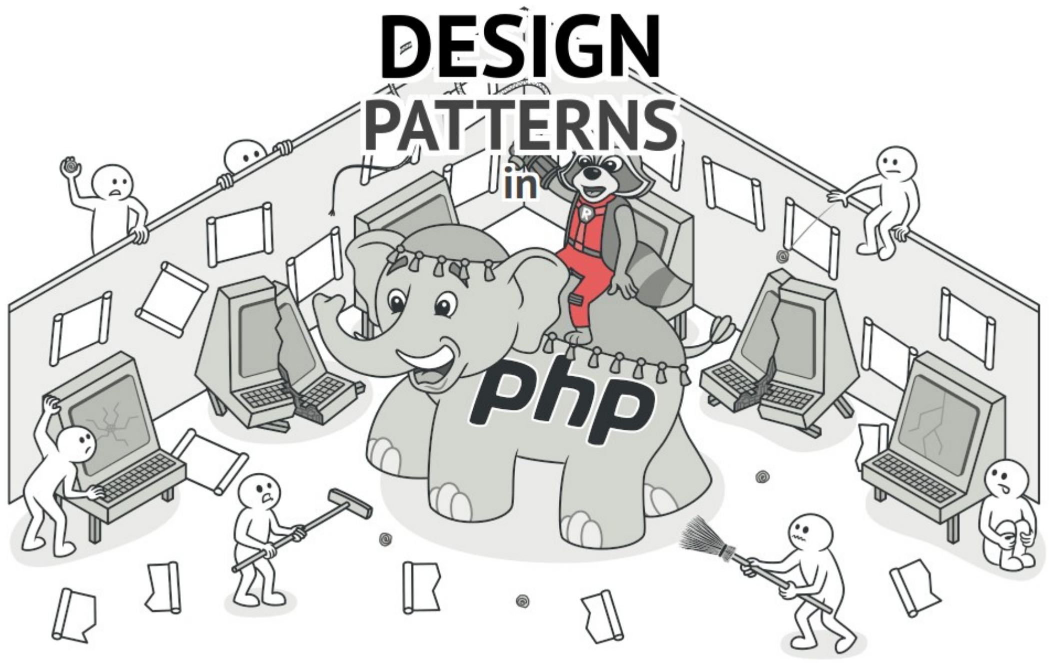
Shop Now!

English Contact us Log in

REFACTORING
· GURU ·

- Premium Content
- Refactoring
- Design Patterns
  - What is a Pattern
  - Catalog
    - Creational Patterns
    - Structural Patterns
    - Behavioral Patterns
  - Code Examples
    - C#
    - C++
    - Go
    - Java
    - → PHP
    - Python
    - Ruby
    - Rust
    - Swift
    - TypeScript

Log in Contact us



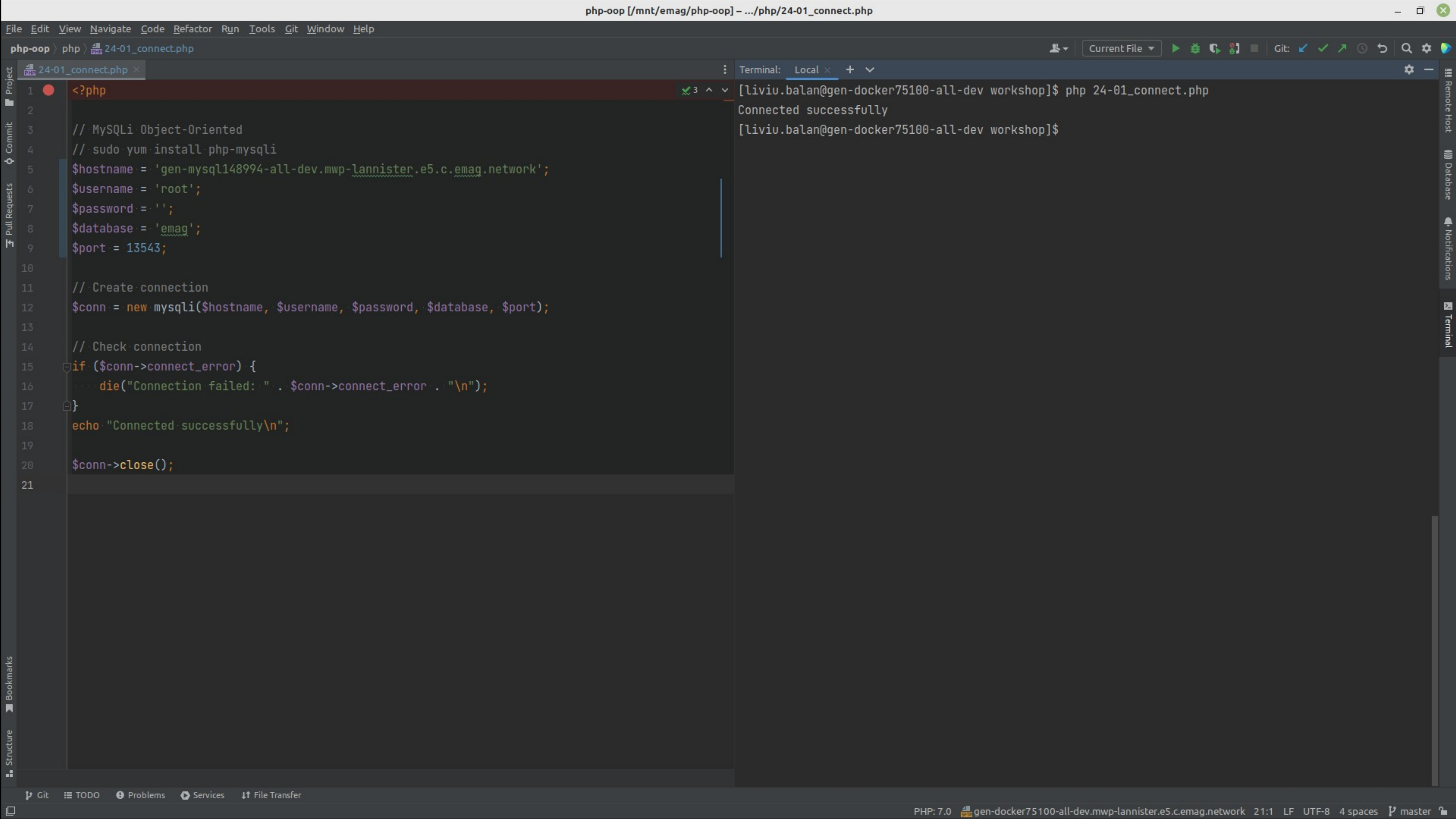
The Catalog of **PHP** Examples

Creational Patterns

Abstract Factory ★★★
Lets you produce families of related objects without specifying their concrete classes.

Structural Patterns

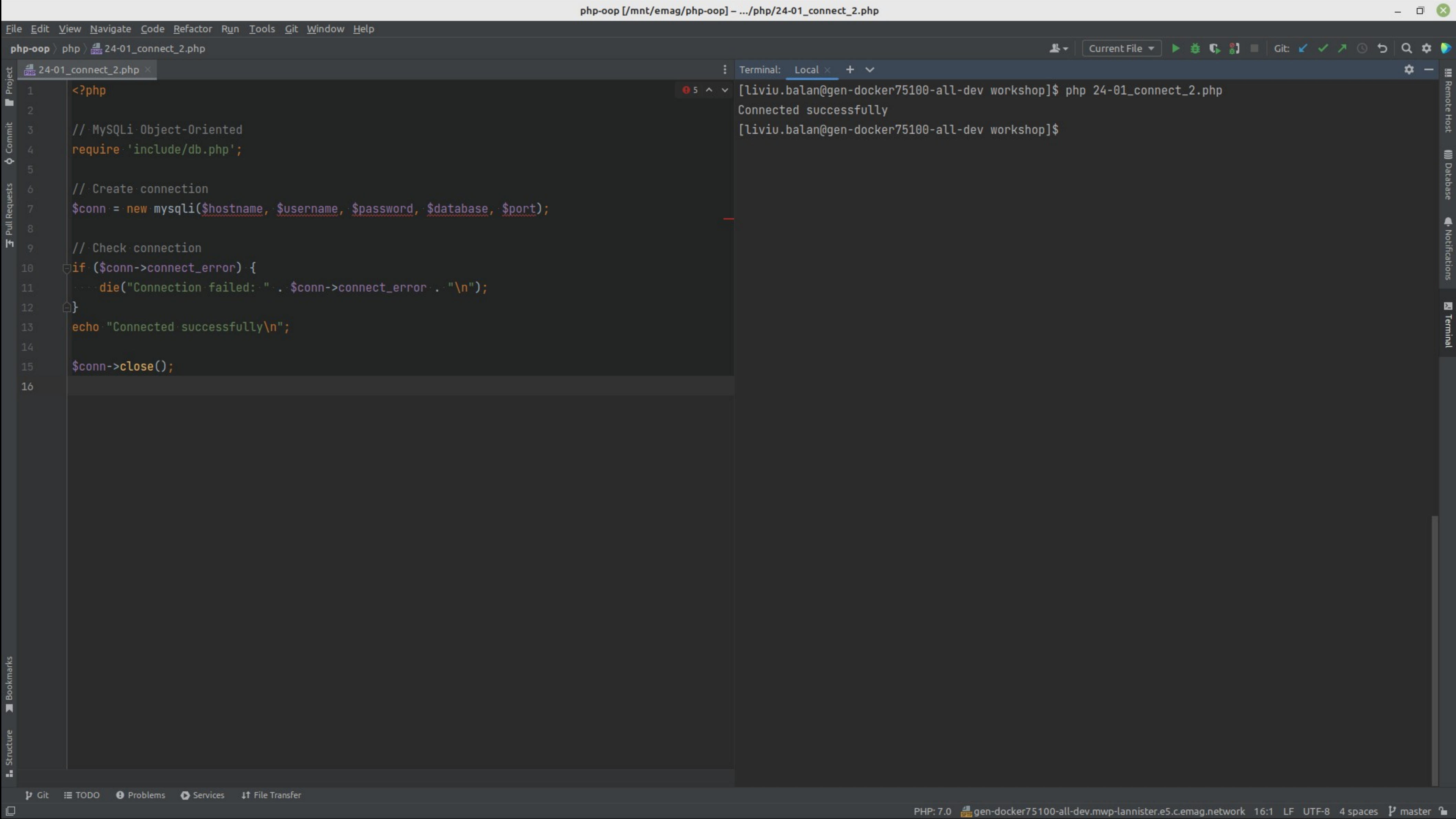
Adapter ★★★
Allows objects with incompatible interfaces to collaborate.



```php
<?php

// MySQLi Object-Oriented
// sudo yum install php-mysqli
$hostname = 'gen-mysql148994-all-dev.mwp-lannister.e5.c.emag.network';
$username = 'root';
$password = '';
$database = 'emag';
$port = 13543;

// Create connection
$conn = new mysqli($hostname, $username, $password, $database, $port);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error . "\n");
}
echo "Connected successfully\n";

$conn->close();
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-01_connect.php
Connected successfully
[liviu.balan@gen-docker75100-all-dev workshop]$
```

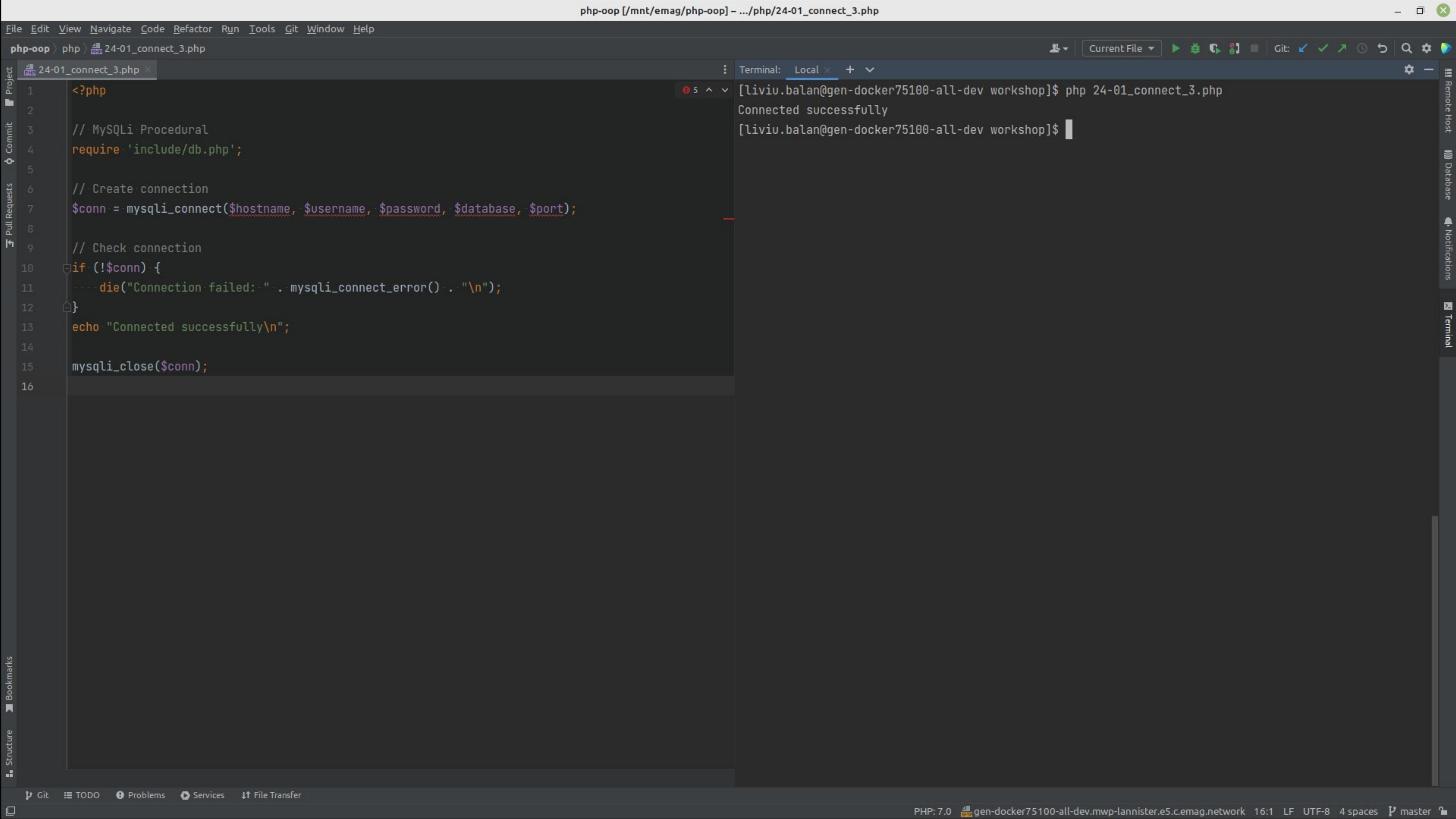
php-oop [/mnt/emag/php-oop] – .../php/24-01_connect.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop php 24-01_connect.php
Current File
Git:
24-01_connect.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 21:1 LF UTF-8 4 spaces master



php-oop [/mnt/emag/php-oop] – .../php/24-01_connect_2.php

```php
<?php

// MySQLi Object-Oriented
require 'include/db.php';

// Create connection
$conn = new mysqli($hostname, $username, $password, $database, $port);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error . "\n");
}
echo "Connected successfully\n";

$conn->close();
```

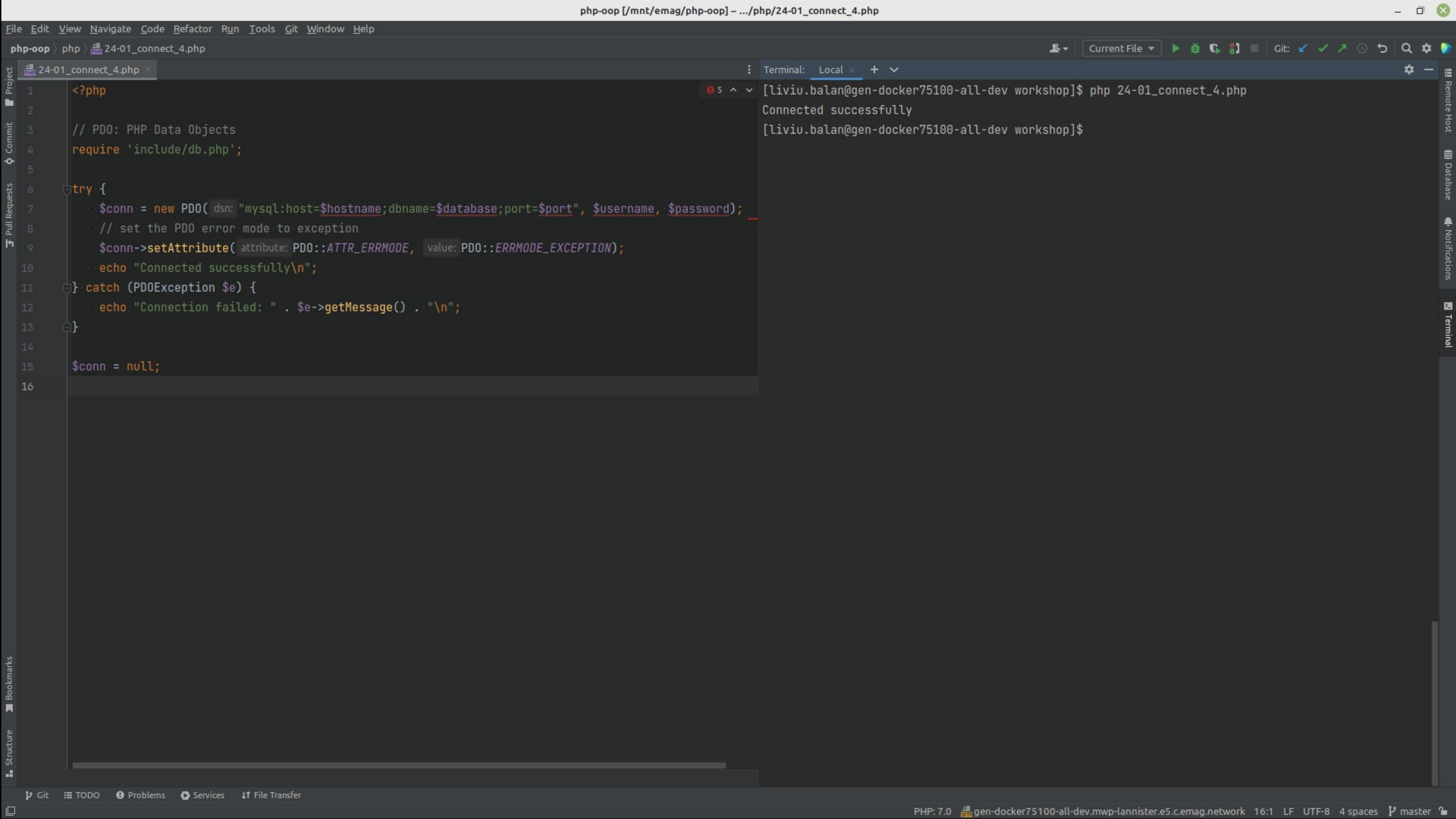
```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-01_connect_2.php
Connected successfully
[liviu.balan@gen-docker75100-all-dev workshop]$
```



```php
<?php

// MySQLi Procedural
require 'include/db.php';

// Create connection
$conn = mysqli_connect($hostname, $username, $password, $database, $port);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error() . "\n");
}
echo "Connected successfully\n";

mysqli_close($conn);
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-01_connect_3.php
Connected successfully
[liviu.balan@gen-docker75100-all-dev workshop]$
```

php-oop [/mnt/emag/php-oop] – .../php/24-01_connect_3.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop php 24-01_connect_3.php
Current File
Git:
24-01_connect_3.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 16:1 LF UTF-8 4 spaces master

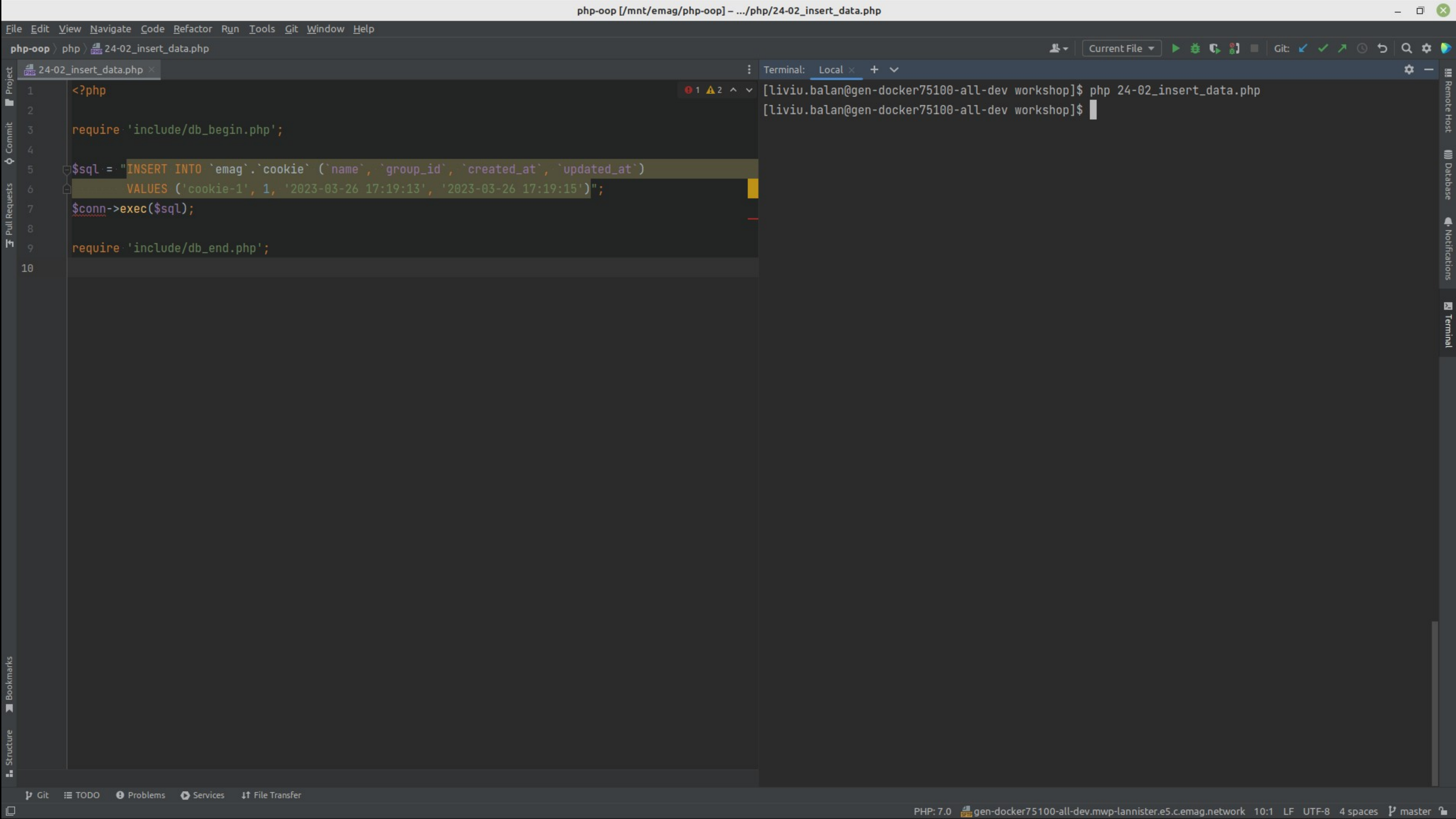


php-oop [/mnt/emag/php-oop] – .../php/24-01_connect_4.php

```php
<?php

// PDO: PHP Data Objects
require 'include/db.php';

try {
    $conn = new PDO( dsn: "mysql:host=$hostname;dbname=$database;port=$port", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute( attribute: PDO::ATTR_ERRMODE,  value: PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully\n";
} catch (PDOException $e) {
    echo "Connection failed: " . $e->getMessage() . "\n";
}

$conn = null;
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-01_connect_4.php
Connected successfully
[liviu.balan@gen-docker75100-all-dev workshop]$
```

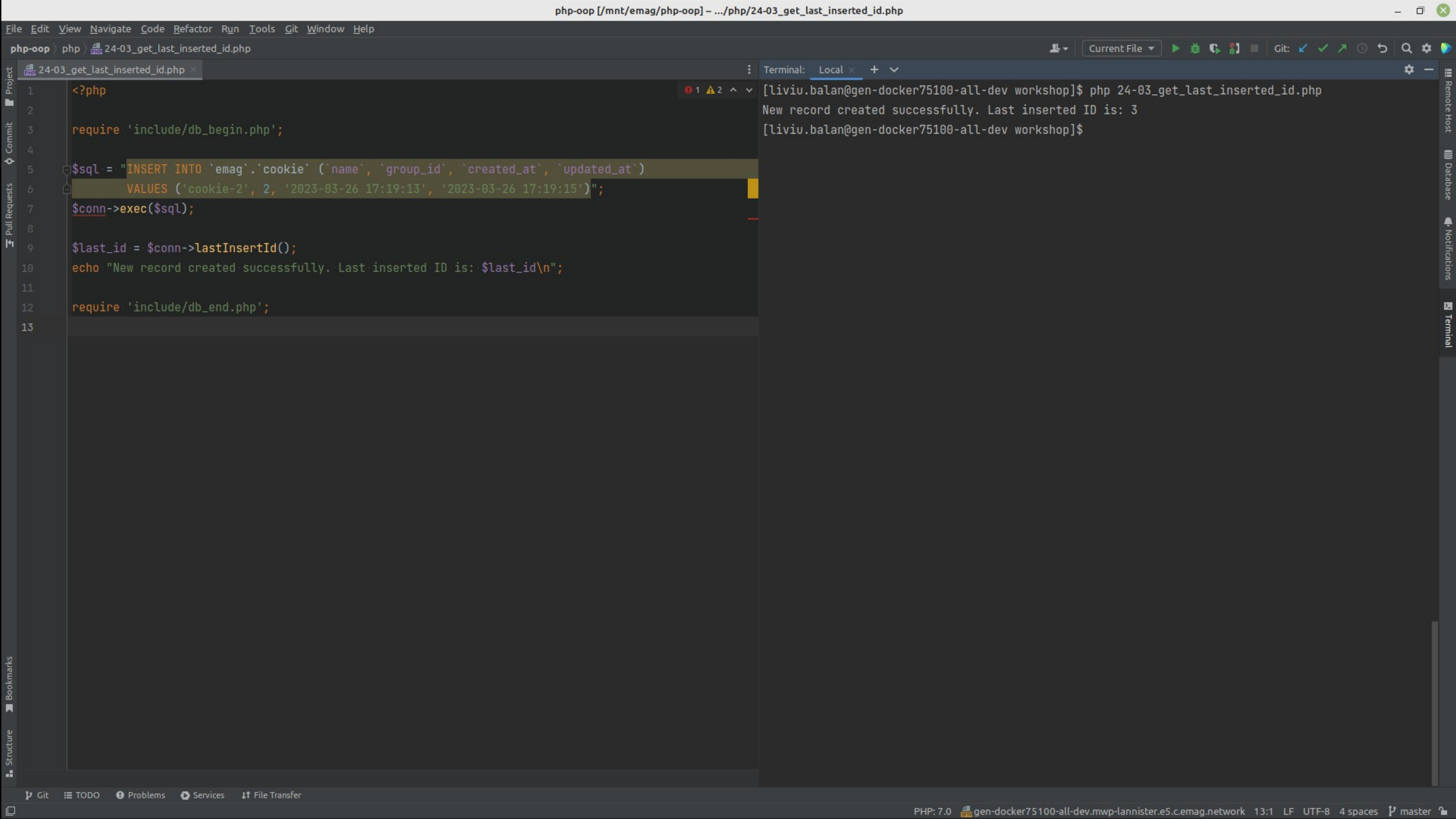
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 16:1 LF UTF-8 4 spaces master



```php
<?php

require 'include/db_begin.php';

$sql = "INSERT INTO `emag`.`cookie` (`name`, `group_id`, `created_at`, `updated_at`)
        VALUES ('cookie-1', 1, '2023-03-26 17:19:13', '2023-03-26 17:19:15')";
$conn->exec($sql);

require 'include/db_end.php';
```

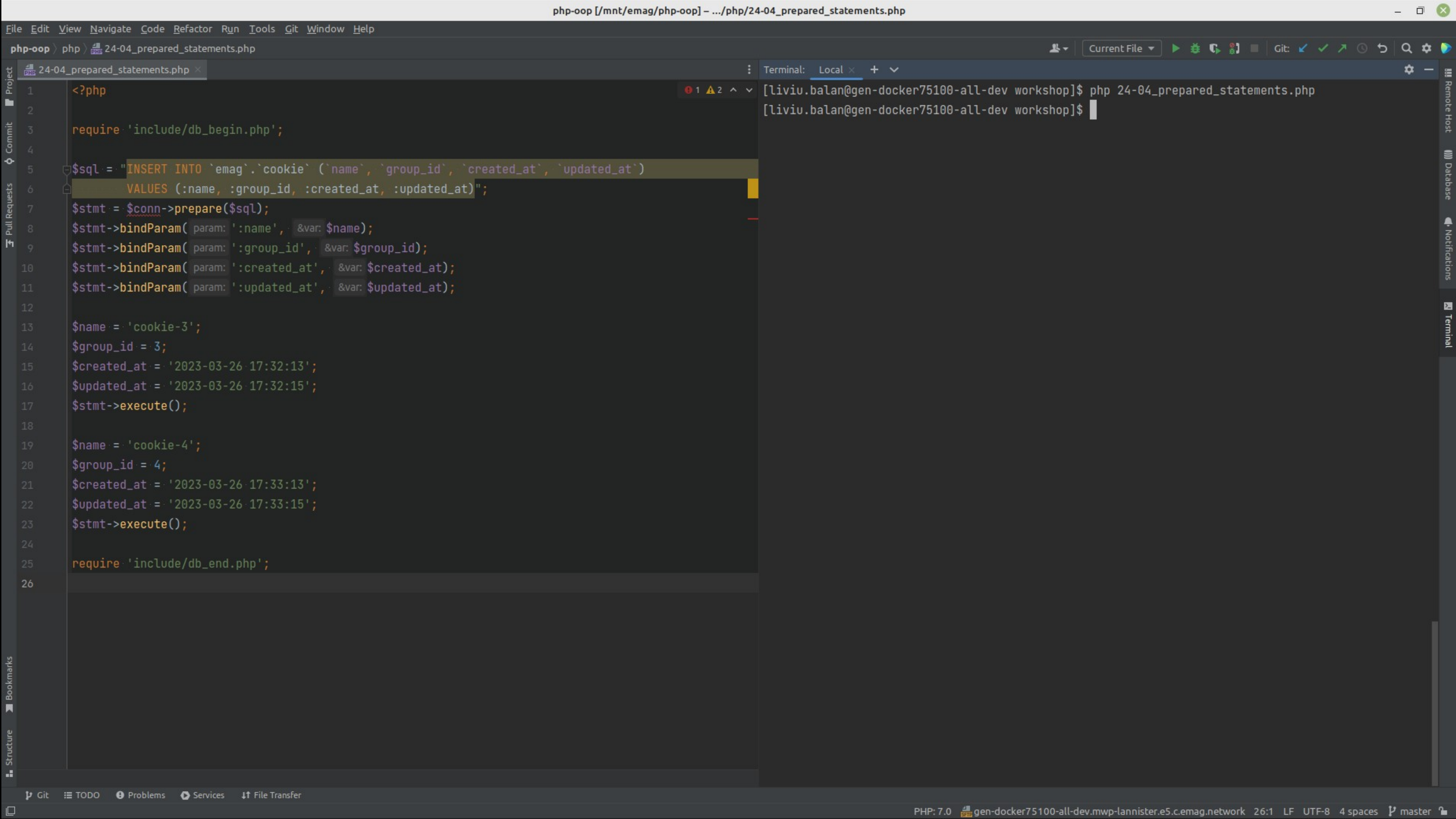
```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-02_insert_data.php
[liviu.balan@gen-docker75100-all-dev workshop]$
```

php-oop [/mnt/emag/php-oop] – .../php/24-02_insert_data.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop php 24-02_insert_data.php
Current File
Git:
24-02_insert_data.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 10:1 LF UTF-8 4 spaces master


php-oop [/mnt/emag/php-oop] – .../php/24-03_get_last_inserted_id.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop php 24-03_get_last_inserted_id.php
Current File
Git:
24-03_get_last_inserted_id.php
Terminal: Local
Git TODO Problems Services File Transfer
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 13:1 LF UTF-8 4 spaces master

```php
<?php

require 'include/db_begin.php';

$sql = "INSERT INTO `emag`.`cookie` (`name`, `group_id`, `created_at`, `updated_at`)
        VALUES ('cookie-2', 2, '2023-03-26 17:19:13', '2023-03-26 17:19:15')";
$conn->exec($sql);

$last_id = $conn->lastInsertId();
echo "New record created successfully. Last inserted ID is: $last_id\n";

require 'include/db_end.php';
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-03_get_last_inserted_id.php
New record created successfully. Last inserted ID is: 3
[liviu.balan@gen-docker75100-all-dev workshop]$
```

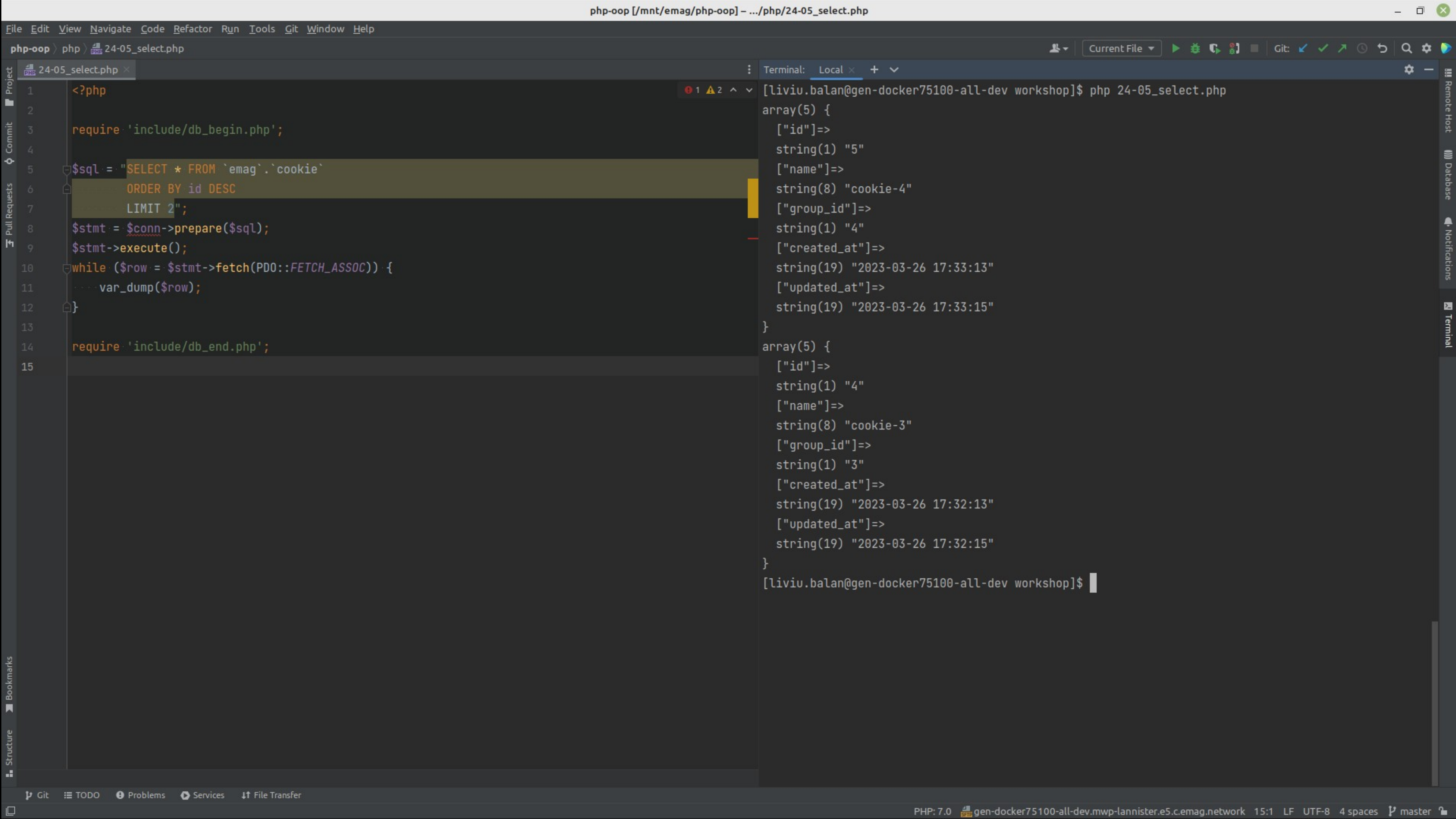


php-oop [/mnt/emag/php-oop] – .../php/24-04_prepared_statements.php

```php
<?php

require 'include/db_begin.php';

$sql = "INSERT INTO `emag`.`cookie` (`name`, `group_id`, `created_at`, `updated_at`)
        VALUES (:name, :group_id, :created_at, :updated_at)";
$stmt = $conn->prepare($sql);
$stmt->bindParam( param: ':name',  &var: $name);
$stmt->bindParam( param: ':group_id',  &var: $group_id);
$stmt->bindParam( param: ':created_at',  &var: $created_at);
$stmt->bindParam( param: ':updated_at',  &var: $updated_at);

$name = 'cookie-3';
$group_id = 3;
$created_at = '2023-03-26 17:32:13';
$updated_at = '2023-03-26 17:32:15';
$stmt->execute();

$name = 'cookie-4';
$group_id = 4;
$created_at = '2023-03-26 17:33:13';
$updated_at = '2023-03-26 17:33:15';
$stmt->execute();

require 'include/db_end.php';
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-04_prepared_statements.php
[liviu.balan@gen-docker75100-all-dev workshop]$
```

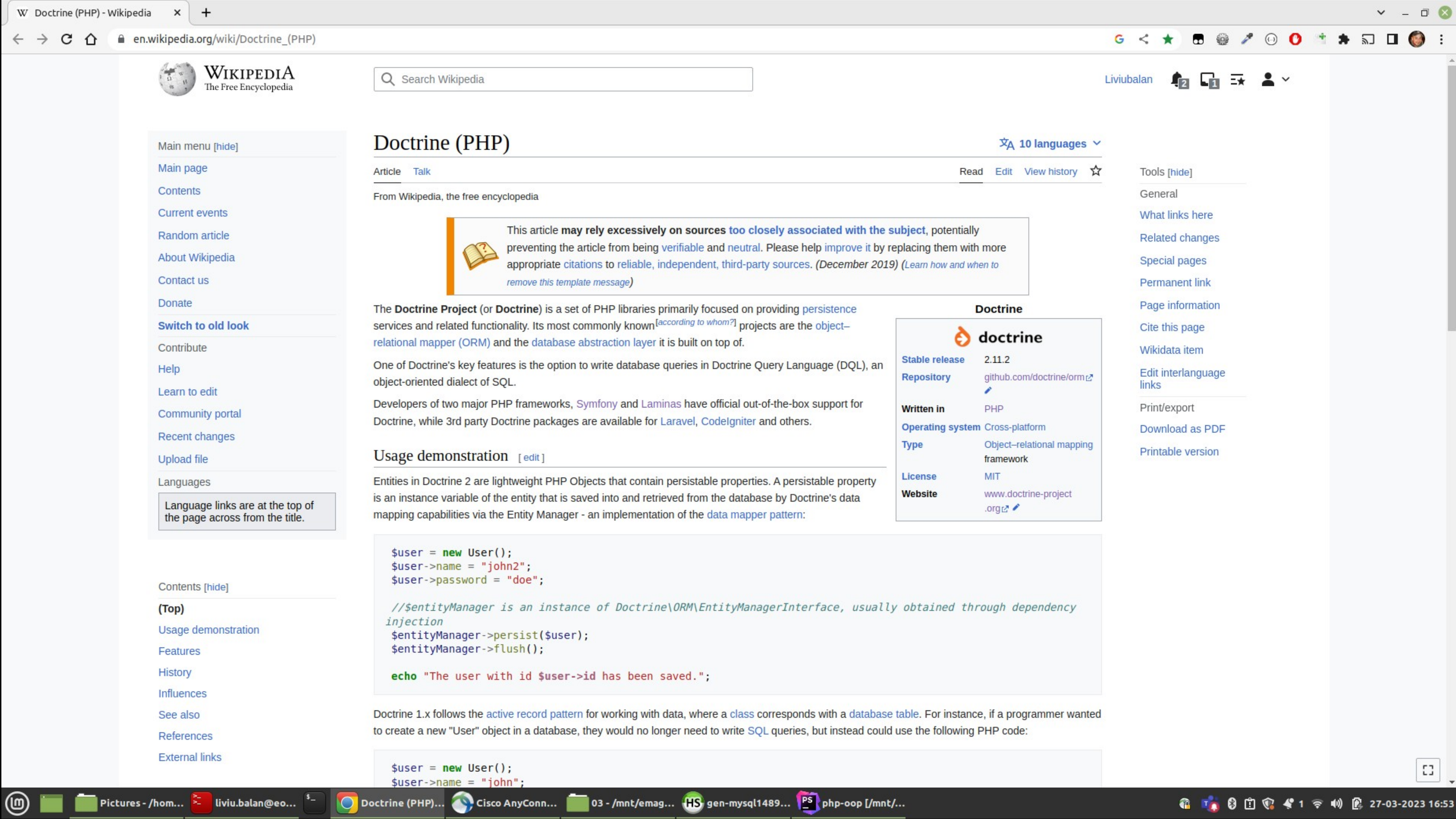
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 26:1 LF UTF-8 4 spaces master



```php
<?php

require 'include/db_begin.php';

$sql = "SELECT * FROM `emag`.`cookie`
        ORDER BY id DESC
        LIMIT 2";
$stmt = $conn->prepare($sql);
$stmt->execute();
while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
    var_dump($row);
}

require 'include/db_end.php';
```

```
[liviu.balan@gen-docker75100-all-dev workshop]$ php 24-05_select.php
array(5) {
  ["id"]=>
  string(1) "5"
  ["name"]=>
  string(8) "cookie-4"
  ["group_id"]=>
  string(1) "4"
  ["created_at"]=>
  string(19) "2023-03-26 17:33:13"
  ["updated_at"]=>
  string(19) "2023-03-26 17:33:15"
}
array(5) {
  ["id"]=>
  string(1) "4"
  ["name"]=>
  string(8) "cookie-3"
  ["group_id"]=>
  string(1) "3"
  ["created_at"]=>
  string(19) "2023-03-26 17:32:13"
  ["updated_at"]=>
  string(19) "2023-03-26 17:32:15"
}
[liviu.balan@gen-docker75100-all-dev workshop]$
```

php-oop [/mnt/emag/php-oop] – .../php/24-05_select.php
File Edit View Navigate Code Refactor Run Tools Git Window Help
php-oop 〉 php 〉 24-05_select.php
Current File
Terminal: Local
PHP: 7.0 gen-docker75100-all-dev.mwp-lannister.e5.c.emag.network 15:1 LF UTF-8 4 spaces master



Doctrine (PHP)

From Wikipedia, the free encyclopedia

This article **may rely excessively on sources too closely associated with the subject**, potentially preventing the article from being verifiable and neutral. Please help improve it by replacing them with more appropriate citations to reliable, independent, third-party sources. *(December 2019)* (*Learn how and when to remove this template message*)

The **Doctrine Project** (or **Doctrine**) is a set of PHP libraries primarily focused on providing persistence services and related functionality. Its most commonly known[according to whom?] projects are the object–relational mapper (ORM) and the database abstraction layer it is built on top of.

One of Doctrine's key features is the option to write database queries in Doctrine Query Language (DQL), an object-oriented dialect of SQL.

Developers of two major PHP frameworks, Symfony and Laminas have official out-of-the-box support for Doctrine, while 3rd party Doctrine packages are available for Laravel, CodeIgniter and others.

Doctrine	
Stable release	2.11.2
Repository	github.com/doctrine/orm
Written in	PHP
Operating system	Cross-platform
Type	Object–relational mapping framework
License	MIT
Website	www.doctrine-project.org

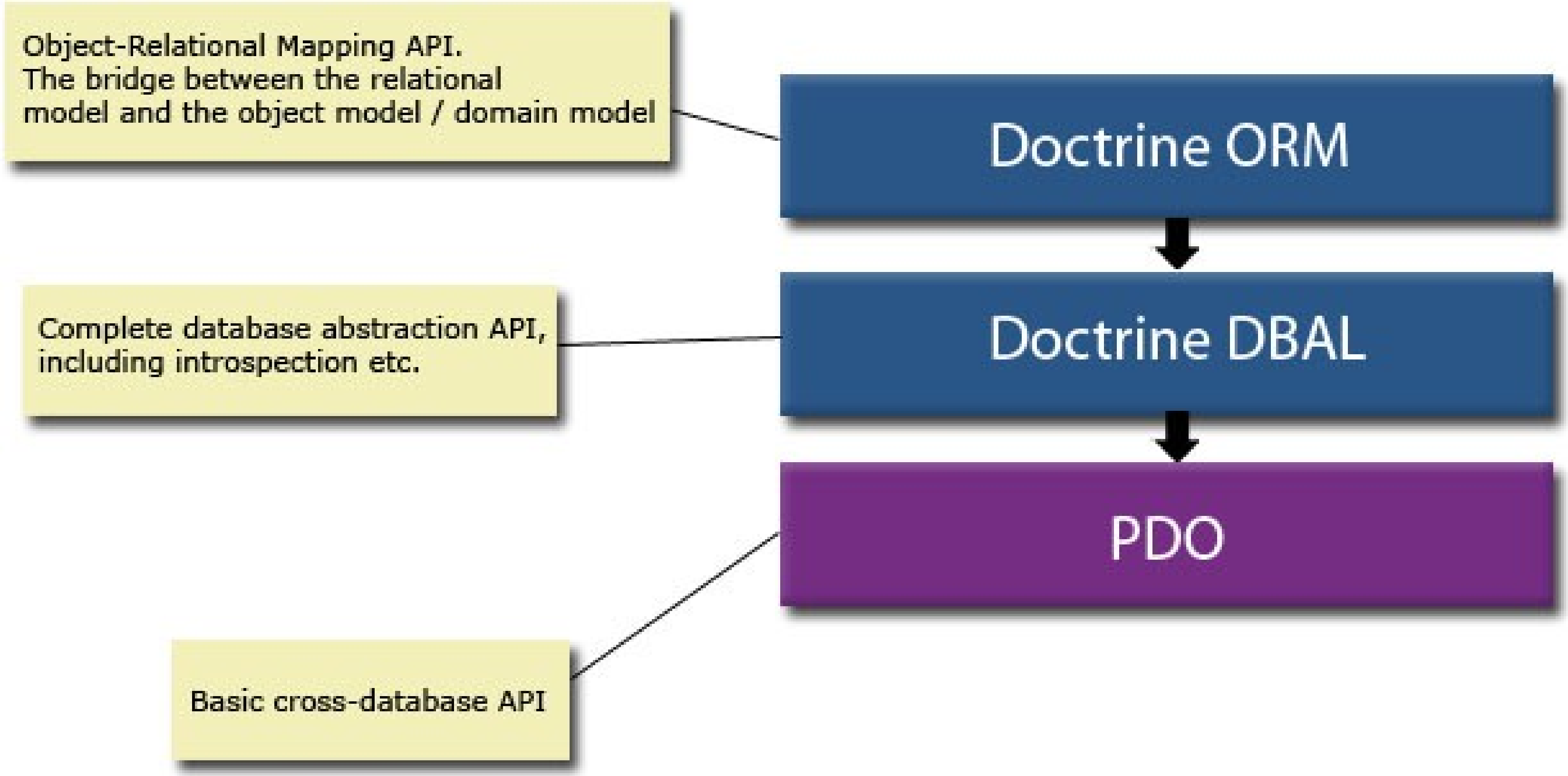
Usage demonstration [edit]

Entities in Doctrine 2 are lightweight PHP Objects that contain persistable properties. A persistable property is an instance variable of the entity that is saved into and retrieved from the database by Doctrine's data mapping capabilities via the Entity Manager - an implementation of the data mapper pattern:

```
$user = new User();
$user->name = "john2";
$user->password = "doe";

//$entityManager is an instance of Doctrine\ORM\EntityManagerInterface, usually obtained through dependency
injection
$entityManager->persist($user);
$entityManager->flush();

echo "The user with id $user->id has been saved.";
```

Doctrine 1.x follows the active record pattern for working with data, where a class corresponds with a database table. For instance, if a programmer wanted to create a new "User" object in a database, they would no longer need to write SQL queries, but instead could use the following PHP code:

```
$user = new User();
$user->name = "john";
```


Object-Relational Mapping API.
The bridge between the relational model and the object model / domain model
Doctrine ORM
Complete database abstraction API, including introspection etc.
Doctrine DBAL
PDO
Basic cross-database API

Q&A

Practice