B2 - Reboost

B-RBT-200

Backend

PHP - Object Oriented Programming

1.1

{EPITECH.}

Backend

repository name: RBT_back_d02_$ACADEMICYEAR
repository rights: anthony1.palasse@epitech.eu fabien-luc1.lallemand@epitech.eu kevin.cazal@epitech.eu nicolas1.ah-leung@epitech.eu william.grondin@epitech.eu

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

+ Introduction

Starting from today, you're going to learn Object Oriented Programming (OOP).
To help you in your researches, here are today's notions (in no particular order) :

- Classes definition
- Objects and instantiations
- Constructors
- Destructors
- Methods and attributes
- New operator
- Methods and attributes visibility
- Optional parameters

+ TASK 01

Turn in: task_01/Gecko.php
Restrictions: None.

Create a new class “Gecko”.

Every times a new Gecko is created, it shall automatically display “Hello !” followed by a newline.

Example:

Terminal

```
~/B-RBT-200> cat example.php
<?php

include_once("Gecko.php");

$alex = new Gecko();
$raph = new Gecko();
```

Terminal

```
~/B-RBT-200> php example.php
Hello !
Hello !
~/B-RBT-200>
```

+ TASK 02

Turn in: task_02/Gecko.php
Restrictions: None.

Copy your "Gecko.php" from Task 01. We're going to add a few things to it.

You shall modify your constructor so that it becomes possible to pass a name as parameter for our new Gecko being created.

If a name is specified, it will now display "Hello <Name> !" followed by a newline.

Otherwise, it will still display the same thing as in task 01.

This parameter will need to be stored in an attribute "name" that will be public.

Example:

Terminal

```
~/B-RBT-200> cat example.php
<?php

include_once("Gecko.php");

$thomas = new Gecko("Thomas");
$annonymus = new Gecko();

echo $thomas->name;
echo $annonymus->name . "\n";
```

Terminal

```
~/B-RBT-200> php example.php | cat -e
Hello Thomas !$
Hello !$
Thomas$
~/B-RBT-200>
```

+ TASK 03

Turn in: task_03/Gecko.php
Restrictions: None.

Once again, copy your "Gecko.php" from the previous task.

From now on, when your Gecko will ceases to exist, you will need to display "Bye <Name> !" followed by a newline.

If they do not have a name, they will simply say "Bye !", followed by a newline.

Ah, speaking of name… From now on, the "name" attribute should not be public, you will need to find yourself what it should be. However, for the name to be available outside our object, we will create our very first method! It will be called "getName" and it will return… the name of the Gecko!

Example:

Terminal

```
~/B-RBT-200> cat example.php
<?php

include_once("Gecko.php");

$thomas = new Gecko("Thomas");
$anonymous = new Gecko();
$serguei = new Gecko("Serguei");

unset($serguei);
echo $thomas->getName() . "\n";
echo $anonymous->getName() . "\n";
```

Terminal

```
~/B-RBT-200> php example.php | cat -e
Hello Thomas !$
Hello !$
Hello Serguei !$
Bye Serguei !$
Thomas$
$
Bye !$
Bye Thomas !$
~/B-RBT-200>
```

+ TASK 04

Turn in: task_04/Gecko.php
Restrictions: None.

Copy your "Gecko.php" from the previous task.

Add a new attribute "Age" to our Gecko class. It should be possible to specify it as a second parameter during the construction of the object.

This attribute will have its own getter and setter, respectively "getAge" and "setAge".

Moreover, you will add a new method "status" to our Gecko. This method will take no parameter, and must display a sentence depending on the Gecko's age. You **must** use a 'switch' statement and you are not allowed to use the 'if' keyword in this method.

The method must display the following sentences:
- "Unborn Gecko" if the age is 0.
- "Baby Gecko" if the age is 1 or 2.
- "Adult Gecko" if the age is between 3 and 10.
- "Old Gecko" if the age is between 11 and 13.
- "Impossible Gecko" otherwise.

Each of these sentences must be followed by a newline.

+ TASK 05

Turn in: task_05/Gecko.php
Restrictions: None.

It is time… to give the gift of speech to our Geckos! Firstly, copy your previous "Gecko.php"

Add a new public method to it, called "hello".

When called with a string, it will display "Hello <string>, I'm <Name>!", with <string> being the string given as parameter and <Name> being the name of the Gecko. If our Gecko doesn't have a name it will only display "Hello <string>!".

However, if an integer is given as parameter it will display
"Hello, I'm <Name>!" as often as the number given as parameter.
Once again, if our Gecko doesn't have a name it will display just "Hello !" as often as the number given as parameter.

Every message must be followed by a newline.

In all other cases, the method does nothing.

Example:

Terminal

```
~/B-RBT-200> cat example.php
<?php

include_once("Gecko.php");

$dylan = new Gecko("Dylan");
$dylan->hello("Teddy");
$dylan->hello(2);
```

Terminal

```
~/B-RBT-200> php example.php | cat -e
Hello Dylan !$
Hello Teddy, I'm Dylan!$
Hello, I'm Dylan!$
Hello, I'm Dylan!$
Bye Dylan !$
~/B-RBT-200>
```

+ TASK 06

Turn in: task_06/Gecko.php
Restrictions: None.

Copy your "Gecko.php" from the previous task.

Add a new method "eat" to our Gecko. It will take a string as parameter and return nothing.

If the value of the string given in parameter is equal to "Meat", the Gecko must display the following: "Yummy!".
If the value is equal to "Vegetable", your Gecko must say "Erk!".
If the value is anything else, the Gecko must say "I can't eat this".
As usual, every sentence will be followed by a newline.

The 'eat' method **must be case insensitive.** So "MEAT", "mEaT", "meat", etc. are considered to be "Meat" and the same rule applies to "Vegetable".

Moreover, you will add a new attribute "energy" to our Gecko. By default it will be equal to 100.

You will add a setter and a getter for this attribute (getEnergy and setEnergy).

Every time our Gecko eats something, he will gain or lose some energy.

If he eats meat he will gain 10 Energy. If he eats vegetable he will lose 10 Energy (A Gecko is carnivorous…).
In all the other cases his energy will not be modified.

A gecko's energy should always be between 0 and 100 (included)

+ TASK 07

Turn in: task_07/Gecko.php
Restrictions: None.

Copy your "Gecko.php" file from the previous task.

You will implement a new method "work" in our Gecko class. It will take no parameter and return nothing.

On every call, if the Gecko has at least 25 energy, it will display "I'm working T.T" followed by a newline, then it will decrease his energy by 9.

If the method is called with less than 25 energy, it will display
“Heyyy… I’m too sleepy, better take a nap!”
followed by a newline, then it will restore 50 energy to our Gecko.