Universidade de Brasília – Faculdade Gama

Técnicas de Programação 1º/2014

Alunos:

Caíque Pereira 13/0104752

Elaine Cristina Meirelles 12/0010551

Larissa Rodrigues 11/0015045

Lucas Carvalho 11/0015762

Professor: Maurício Serrano

Code Conventions

❖ Indentation

The indentation should be 4 spaces. Every time the level is lowered, should give 4 spaces.

❖ Maximum line length

There may be 100 characters per line of code, always providing the best understanding of the code. If the line length exceed the maximum permitted and has several clauses, it's recommended break the line into multiple lines. In such a case, break the line after to a logic operator, and rearrange the line such that it aligns under the first character of the parameters or conditionals in the previous line.

```
public function listarTodos(){
        $sql = "SELECT * FROM time";
        $resultado = $this->conexao->banco->Execute($sql);
        while($registro = $resultado->FetchNextObject()){
                $dadosTime = new Time();
                $dadosTime->__constructOverload($registro->ID_TIME, $registro->TECNICO_ID_TECNICO,
                                                $registro->NOME,$registro->CATEGORIA,
                                                $registro->ENDERECO, $registro->DATA_FUNDACAO,
                                                $registro->PRESIDENTE, $registro->TELEFONE);
                $retornaTime[] = $dadosTime;
        }
        return $retornaTime;
}
```

❖ Header

The header must contain the name and description of the class. There should be one break line before this. Also, before 'name' and 'description' should have indentation, as well as the subsequent lines, if is more than one line in any of the fields. The "/*" and "*/" should be in the most left side, with no spaces. Moreover, all the phrases have to end with a dot, except the line about the name of the class. For example:

```php
<?php

/*
    Name: Arbitro.php
    Description: This is the class that contains the attributes, accessor methods
    and constructors of a referee. A referee must contains 'idArbitro', 'nome',
    'telefone' and 'cpf'.
 */
class Arbitro
```

❖ Comments

All comments should use /* ... */ and they should be aligned with the function or method associated. The comments will be in english, as the PHP language. Before the comments, there should be four spaces after the method or function identation. It should start with capital letter and end with dot. The comments have to be one line before the method or function declaration. There should be one blank line before the /*.

```php
    /*
        Method to modify the attribute 'idAtributo' of the class.
    */
    public function   setIdArbitro($idArbitro){
```

❖ Control structures

The braces, "{" and "}", have to be opened in the same line that the class or the conditional of control structures, like 'for', 'while', 'if', 'else', begins and in it is own line when the code is finished. About the code, it has to be in the next line after the opening brace.

```
for($i = 0; $i < count($arrayDadosJogador); $i++){
        $dadosJogador = $arrayDadosJogador[$i];
        if($dadosJogador -> __getIdTime() == $time){
            $jogadorTime[] = $dadosJogador;
        }
}
```

❖ Empty functions or methods or control structures

When the function or methods or control structures is empty, after the opening brace have to be a break line before the ending brace.

```
public function __construct(){

}
```

❖ Functions and variables names

The names of functions and variables must begin with lowercase letters, and if they are compound names should be the next words with the first letter capitalized. The use of dots or underscores in any part of the name is not allowed. In addition, the names cannot begin with capital letters.

```
private $idArbitro;
```

❖ Packages name

The package names should always be related to the content and in english. Also, it should start with lower case letter. The use of underscore and capital words in any part of the name is not allowed. It's possible to use dots in the middle of the name, never in the beginning or the end.

❖ Spaces:

- ✓ **Attribution:** It is necessary to use one space before and after the assignment.

```
$idTime = 0945;
```

✓ **Operations:** The operations (-, +, =, *, <, >) should have one space before and after the operator, except the operator "->", this should not have the space.

```
/* Correct */
$a = $a1 + $a2;

/* Incorrect */
$a=$a1+$a2;
```

✓ **Parameters:** It should use one space after each comma.

```
public function _atualizar($idJogo, $idTime){
    $dadosTimeJogo = new TimeJogo();
    $dadosTimeJogo->__constructOverload($idJogo, $idTime);
    $this->timeJogoDAO->atualizar($dadosTimeJogo);
}
```

✓ **Function or methods:** Do not put a space following a function name or between the final parentheses of the parameters attribution and the opening brace.

```
public function excluir($id){
        $sql = "DELETE FROM time WHERE id_time = '{$id}' ";
        $resultado = $this->conexao->banco->Execute($sql);
}
```

✓ **Parentheses**: It is not necessary to put spaces next to parentheses or square brackets or braces on the inside, even in structures like 'if', 'elseif', 'switch', 'while'.

```
/* Correct */
$arrayDados['total7Metros'] = $dadosJogo->__getTotal7Metros();

/* Incorrect */
$arrayDados[ 'total7Metros' ] = $dadosJogo->__getTotal7Metros();
```

✓ **Control structures:** Control structures like 'if', 'for', 'while', 'switch', etc. do not

have to be followed by a space. Also, there is no space between the final parentheses, after the conditional, and the opening brace.

```
for($i = 0; $i < count($arrayDadosJogador); $i++){
        $dadosJogador = $arrayDadosJogador[$i];
        if($dadosJogador -> __getIdTime() == $time){
                $jogadorTime[] = $dadosJogador;
        }
}
```

❖ Constants

- ✓ Constants may contain both alphanumeric characters and underscores. Numbers are permitted in constant names.
- ✓ All letters used in a constant name must be capitalized, while all words in a constant name must be separated by underscore characters.

EMBED_SUPPRESS_EMBED_EXCEPTION

- ✓ Defining constants in the global scope with the 'define' function is permitted. But, constants can be defined as class members with the 'const' modifier.

❖ PHP Code Demarcation

PHP code must always be delimited by the full-form, standard PHP form. It should not close the php.

```
<?php
.
.
.
.
```

❖ Commits

The commit description must follow this model: “action action-description file[technique used]”.

Added functions comments to view/TimeView.php [style and design]

Ps.: The most of this rules can be found in the url example below:
https://gist.github.com/Lcunha/9834402