W12 – PHP – Web Programming III

POO in PHP

class

```php
class Character {
 //Constants always public
 const POWER_MAGICIAN = 45;
 //Constants witten in all cap
 const POWER_THIEF = 80;
 const POWER_BARBARIAN = 250;

 //can be public or private
 private $_name;
 //attributes always start with _
 private $_age; private $_force; private $_hp;

 //initialization possible but with static values
 private $_intiGood = "hello";
 private $_initBad = 47 + 4;

 private function addHp($hpToAdd) {
    $this->_hp += $hpToAdd;
 }

 public function removeHp($hpToRemove) {
    $this->_hp -= $hpToRemove;
 }

 public function hit(Character $trarget) {
    $trarget->removeHp($this->_force);
 }
}
```

__ constructor

```php
class Student {
	//attributes
	private $_name; private $_age;

	//constructor – can have only one constructor
	public function __construct($_name, $_age) {
		$this->_name = $_name;
		$this->_age = $_age;
	}

	//getter setter
	public function getname() {
		return $this->_name;
	}

	public function setname($name) {
		if (strlen($name) < 4)
			trigger_error("name is too short", E_USER_WARNING);
		else
			$this->_name = $name;
	}
	public function getAge() {
		return $this->_age;
	}
	public function setAge($age) {
		if (!is_int($age)) {
			trigger_error("enter valid age", E_USER_ERROR);
			return;
		} else {
			$this->_age = $age;
		}
	}
}
```

Going further with PHP

__get __set __toString class methods

- using interfaces
- using exception handling
- using inheritance

And many more...

instanciation

```php
<?php
//to have access to a class
require "Student.php";

$stud = new Student();
$stud2 = new Student("jf", "lidou", 36);

//use -> to call methods of a class
$perso = new Character();
$perso->removeHp(10);
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<h1>Hello <?=$stud2->getname()?> </h1>
<h1>A character constant : <?
=Character::POWER_MAGICIAN?></h1>
</body>
</html>
```

static

```php
class StudentManager {
        const HELLO = "test hello";
        private static $queryGetAll = "select * from
student";
        public static $queryGetById = "select * from
student where id = ?";

//specify the return type
    public static function getAll() : array {
        //connect to the db
        return [new Student(), new Student()];
    }
    public static function getById($id) : Student
{
        if(!is_int($id)){
                return null;
                //use self to call constants and
static methods from a class
                echo self::HELLO;
        }else{
                return [new Student()];
        }
    }
}
```

```php
//Use :: to call methods, const,
or static attributes
$stud =
StudentManager::getAll();
$valueStatic = StudentManager::
$queryGetById;
echo StudentManager::HELLO
```

serialize() / unserialize()

```php
$tab = ["test", 4 , true, 56.45];
$serialisTab = serialize($tab);

echo $serialisTab;
//a:4:
{i:0;s:4:"test";i:1;i:4;i:2;b:1;i:3;d:56.450
000000000003;}

echo "<br />";
var_dump(unserialize($serialisTab));
//array(4)    [0]=> string(4) "test" [1]=>
{ int(4)      bool(true) [3]=> float(56.45) }
[2]=>
```

```php
$user = new User(1, "stud", "abc" );

$serialObj  = serialize($user);
echo $serialObj  .'<br />';
//O:4:"User":3:
{s:9:"User_id";i:1;s:12:"User_login";s:
3:"stud";s:10:"User_pwd";s:3:"abc";}

$unserialObj = unserialize($serialObj);
echo $unserialObj;
//1 : stud : ab
```

```php
class User{
    private $_id; private $_login; private $_pwd;
    public function __construct(int $_id,string
    $_login,string $_pwd) {
        $this->_id = $_id;
        $this->_login = $_login;
        $this->_pwd = $_pwd;
    }
    public function getId() {
        return $this->_id;
    }
    public function setId($id) {
        $this->_id = $id;
    }
    public function getLogin() {
        return $this->_login;
    }
    public function setLogin($login) {
        $this->_login = $login;
    }
    public function getPwd() {
        return $this->_pwd;
    }
    public function setPwd($pwd) {
        $this->_pwd = $pwd;
    }
    //php method to override the print of an object
    public function __toString() {
      return $this->_id . ' : ' . $this->_login .
      ' :' . $this->_pwd;
}
}
```

Storing objects in a session

```php
session_start();
$userSession = new User(45, "marcel", "45687");
//saving objects
$_SESSION['user'] = serialize($userSession);
//accessing saved objects
$myUserInSession = unserialize($_SESSION['user']);
```