

# PHP : Exception

**Achref El Mouelhi**

Docteur de l'université d'Aix-Marseille  
Chercheur en programmation par contrainte (IA)  
Ingénieur en génie logiciel

`elmouelhi.achref@gmail.com`



# Plan

- 1 Introduction
- 2 Capture d'exception
- 3 Exceptions personnalisées
- 4 Multi-catch
- 5 Exceptions paramétrées
- 6 Bloc `finally`

## Exception ?

- Erreur qui se produit pendant l'exécution
- Impliquant généralement l'arrêt d'exécution du programme

## Comment faire pour poursuivre l'exécution ?

- Repérer les blocs pouvant générer une exception
- Capturer l'exception correspondante
- Afficher un message relatif à cette exception
- Continuer l'exécution

Considérons la fonction `division` définie dans `index.php`

```
function division($var1, $var2)
{
    return $var1 / $var2;
}
```

© Achref EL MOUELHI ©

Considérons la fonction `division` définie dans `index.php`

```
function division($var1, $var2)
{
    return $var1 / $var2;
}
```

Appeler la fonction `division` avec les paramètres suivants génère plusieurs erreurs

```
$x = 'a';
$y = 0;
echo division($x / $y);
```

Considérons la fonction `division` définie dans `index.php`

```
function division($var1, $var2)
{
    return $var1 / $var2;
}
```

Appeler la fonction `division` avec les paramètres suivants génère plusieurs erreurs

```
$x = 'a';
$y = 0;
echo division($x / $y);
```

### Le message affiché à l'exécution

```
Warning: A non-numeric value encountered in
C:\wamp64\www\cours-poo\index.php on line 16
Warning: Division by zero in C:\wamp64\www\cours-poo\index.php on line
16
NAN
```

## Comment faire pour capturer une exception ?

- Utiliser `throw` pour lancer un objet de type `Exception` contenant le message de l'exception
- Utiliser un bloc `try { ... }` pour entourer une instruction utilisant une instruction qui peut lancer une exception
- Utiliser un bloc `catch { ... }` pour capturer l'exception et afficher le message envoyé



## Modifions la déclaration de la fonction `division`

```
function division($var1, $var2)
{
    if ($var2 == 0){
        throw new Exception("Le deuxième paramètre
            doit être différent de 0");
        die();
    }
    if (!is_numeric($var1) || !is_numeric($var2)){
        throw new Exception("Les deux paramètres
            doivent être de type numérique");
        die();
    }
    return $var1 / $var2;
}
```

**Modifions l'appel de la fonction `division` en ajoutant les blocs `try` et `catch`**

```
$x = 'a';  
$y = 0;  
try {  
    echo division($x, $y);  
} catch (Exception $e) {  
    echo "Problème avec le.s paramètre.s";  
}
```

© Achref EL MOUELHI

**Modifions l'appel de la fonction `division` en ajoutant les blocs `try` et `catch`**

```
$x = 'a';  
$y = 0;  
try {  
    echo division($x, $y);  
} catch (Exception $e) {  
    echo "Problème avec le.s paramètre.s";  
}
```

Le message affiché à l'exécution

Problème avec le.s paramètre.s

## Modifions l'appel de la fonction `division` en ajoutant les blocs `try` et `catch`

```
$x = 'a';  
$y = 0;  
try {  
    echo division($x, $y);  
} catch (Exception $e) {  
    echo "Problème avec le.s paramètre.s";  
}
```

Le message affiché à l'exécution

Problème avec le.s paramètre.s

### Constatation

- Le message qu'on a défini au lancement de l'exception n'a pas été affiché
- Il faut le récupérer

## Pour afficher le message envoyé au lancement de l'exception

```
$x = 'a';  
$y = 0;  
try {  
    echo division($x, $y);  
} catch (Exception $e) {  
    echo echo $e->getMessage();  
}
```

© Act

## Pour afficher le message envoyé au lancement de l'exception

```
$x = 'a';  
$y = 0;  
try {  
    echo division($x, $y);  
} catch (Exception $e) {  
    echo $e->getMessage();  
}
```

Le message affiché à l'exécution

Le deuxième paramètre doit être différent de 0

Pour afficher le deuxième message, modifions la deuxième valeur

```
$x = 'a';  
$y = 2;  
try {  
    echo division($x, $y);  
} catch (Exception $e) {  
    echo $e->getMessage();  
}
```

© Act

# PHP

Pour afficher le deuxième message, modifions la deuxième valeur

```
$x = 'a';  
$y = 2;  
try {  
    echo division($x, $y);  
} catch (Exception $e) {  
    echo $e->getMessage();  
}
```

Le message affiché à l'exécution

Les deux paramètres doivent être de type numérique



## Remarques

- **PHP** nous permet de définir des exceptions personnalisées
- Une exception personnalisée est une classe qui hérite de la classe `Exception`

## Considérons la classe Adresse suivante

```
namespace Models;

class Adresse
{
    public function __construct(
        private string $rue,
        private string $codePostal,
        private string $ville)
    {
    }

    // + les getters/setters et __toString
}
```

Supposons que

`codePostal` doit contenir exactement 5 chiffres (ou caractères)

© Achref EL MOUELHI

## Supposons que

`codePostal` doit contenir exactement 5 chiffres (ou caractères)

## Démarche à faire

- Créer notre classe exception (qui doit étendre la classe `Exception`)
- Dans le constructeur de `Adresse`, on lance une exception si `codePostal` ne contient pas 5 chiffres

**Créons l'exception** `IncorrectCodePostalException` **dans un répertoire**  
`exceptions`

```
namespace exceptions;

class IncorrectCodePostalException extends \Exception
{
    public function __construct()
    {
        echo "Le code postal doit contenir 5 caractères.";
    }
}
```

## Modifions le constructeur de la classe Adresse

```
namespace Models;

use exceptions\IncorrectCodePostalException;

class Adresse
{
    public function __construct(
        private string $rue,
        private string $codePostal,
        private string $ville)
    {
        if (strlen($codePostal) != 5) {
            throw new IncorrectCodePostalException();
            die();
        }
    }

    // le reste ne change pas
}
```

Testons tout cela dans `index.php`

```
include "models/Adresse.php";
include "exceptions/IncorrectCodePostalException.php";

use exceptions\IncorrectCodePostalException;
use models\Adresse;

$a = null;
try {
    $a = new Adresse ("rue de paradis", "1300", "Marseille");
}
catch (IncorrectCodePostalException $icpe) {
    $icpe->getMessage();
}
```

# PHP

Testons tout cela dans `index.php`

```
include "models/Adresse.php";
include "exceptions/IncorrectCodePostalException.php";

use exceptions\IncorrectCodePostalException;
use models\Adresse;

$a = null;
try {
    $a = new Adresse ("rue de paradis", "1300", "Marseille");
}
catch (IncorrectCodePostalException $icpe) {
    $icpe->getMessage();
}
```

Le message affiché

Le code postal doit contenir exactement 5 chiffres



## On peut rajouter une deuxième condition

- `codePostal` doit contenir exactement 5 chiffres
- `rue` doit être une chaîne en majuscule

**Créons une deuxième exception** `IncorrectStreetNameException` **dans le répertoire** `exceptions`

```
namespace exceptions;

class IncorrectStreetNameException extends \Exception
{
    public function __construct()
    {
        echo "Le nom de la rue doit être en majuscule.";
    }
}
```

## Modifions le constructeur de la classe Adresse

```
public function __construct(  
    private string $rue,  
    private string $codePostal,  
    private string $ville)  
{  
    if (strlen($codePostal) != 5) {  
        throw new IncorrectCodePostalException();  
        die();  
    }  
  
    if (strtoupper($rue) != $rue) {  
        throw new IncorrectStreetNameException();  
        die();  
    }  
}
```

# PHP

Re-testons tout cela dans `index.php`

```
$a = null;
try {
    $a = new Adresse ("paradis", "13000", "Marseille");
}
catch (IncorrectCodePostalException $icpe) {
    $icpe->getMessage();
}
catch (IncorrectStreetNameException $isne) {
    $isne->getMessage();
}
```

# PHP

Re-testons tout cela dans `index.php`

```
$a = null;
try {
    $a = new Adresse ("paradis", "13000", "Marseille");
}
catch (IncorrectCodePostalException $icpe) {
    $icpe->getMessage();
}
catch (IncorrectStreetNameException $isne) {
    $isne->getMessage();
}
```

Le message affiché

Le nom de la rue doit être en majuscule

Depuis PHP 7.1, on peut fusionner les `catch`

```
$a = null;
try
{
    $a = new Adresse ("paradis", "13000", "Marseille");
    echo $a;
}
catch (IncorrectCodePostalException | IncorrectStreetNameException $e)
{
    $e->getMessage();
}
```

## Question

Comment faire si on veut afficher les valeurs qui ont déclenché l'exception dans le message ?

**Modifions la première exception** `IncorrectCodePostalException`

```
class IncorrectCodePostalException extends \Exception
{
    public function __construct(string $cp)
    {
        echo("Le code postal '$cp' doit contenir 5 caractères");
    }
}
```



**Modifions la deuxième exception** `IncorrectStreetNameException`

```
class IncorrectStreetNameException extends \Exception
{
    public function __construct(string $rue)
    {
        echo "Le nom de la rue '$rue' doit être en majuscule";
    }
}
```

**Modifions le constructeur de la classe** Adresse

```
public function __construct(  
    private string $rue,  
    private string $codePostal,  
    private string $ville)  
{  
    if (strlen($codePostal) != 5) {  
        throw new IncorrectCodePostalException($codePostal);  
        die();  
    }  
  
    if (strtoupper($rue) != $rue) {  
        throw new IncorrectStreetNameException($rue);  
        die();  
    }  
}
```

## Pour tester

```
$a = null;
try
{
    $a = new Adresse ("paradis", "1300", "Marseille");
}
catch (IncorrectCodePostalException | IncorrectStreetNameException $e)
{
    $e->getMessage();
}
```

© Achille

## Pour tester

```
$a = null;  
try  
{  
    $a = new Adresse ("paradis", "1300", "Marseille");  
}  
catch (IncorrectCodePostalException | IncorrectStreetNameException $e)  
{  
    $e->getMessage();  
}
```

## Message affiché

Le code postal '1300' doit contenir 5 caractères.

## Exercice

Créer une nouvelle classe d'exception `AdresseException` pour fusionner et remplacer les deux exceptions `IncorrectCodePostalException` et `IncorrectStreetNameException`

À utiliser lorsqu'on veut exécuter une instruction qu'une exception soit levée ou non.

## Exemple

```
$a = null;
try
{
    $a = new Adresse("paradis", "13000", "Marseille");
}
catch (IncorrectCodePostalException | IncorrectStreetNameException $e)
{
    $e->getMessage();
} finally {
    echo "Instruction exécutée systématiquement";
}
```

## Remarque

Le bloc `finally` peut s'avérer intéressant si le `catch` contient un `return` qui forcera l'arrêt de l'exécution du code. Malgré cela, ce bloc (`finally`) sera exécuté.