La gestion des erreurs en PHP

C. BENSARI

Introduction

- Un programme est susceptible de rencontrer une situation anormale qui peut l'empêcher de fonctionner
- Ces situations anormales peuvent provenir de différentes manières. Exemples :
  - Erreur de la part du développeur dans la syntaxe du langage
  - Erreur lors de l'exécution du programme suite à l'inclusion d'un fichier qui n'existe pas
- Dans ces situations, PHP envoi ce qu'on appelle une «erreur» qui se compose de deux informations :
  - Le type d'erreur souvent sous forme de constante (nom et valeur)
  - Le message d'erreur

Les niveaux d'erreurs PHP

- En PHP il existe plusieurs niveaux (types) d'erreurs, les plus connues sont :
  - E_NOTICE : erreur de type information, elle ne bloque pas l'exécution de la suite du script PHP
  - E_WARNING : erreur de type avertissement, elle ne bloque pas l'exécution de la suite du script PHP
  - E_ERROR : erreur fatale, elle bloque systématiquement l'exécution du script

Exemple

```php
<?php
    echo $a; // affichera erreur de type notice ..le script continue

    // Le fichier inconnu.php étant pas existant
    include ('inconnu.php'); // affichera une erreur de type warning, le script continu

    require('inconnu.php');  // affiche  erreur fatale, le script s'arrête

    echo  'hello' ;   // ne sera pas exécuté
?>
```

Création de gestionnaire d'erreurs :

- Dans la plupart des cas, lorsqu'une erreur est envoyée par PHP, il est préférable de ne pas l'afficher dans son état brut à l'utilisateur final de l'application
- PHP définit une fonction appelée « set_error_handler » qui peut prendre deux paramètres :
  - Le premier paramètre est une **« fonction anonyme »** qui sera exécutée lorsque PHP envoi une erreur
  - Le deuxième paramètre (optionnel) est le niveau de l'erreur pour lequel la fonction anonyme sera exécutée
- La fonction anonyme prend au minimum 2 paramètres (niveau d'erreur et le message d'erreur)et deux paramètres optionnels (nom du fichier où s'est produite l'erreur et le numéro de la ligne)

Exemple

```
<?php
   set_error_handler(function($niveau, $message, $fichier, $ligne){
            // exemple : un affichage personnalisé de l'erreur
            echo « <strong> Niveau: $niveau </strong><br/>";
            echo " Message: $message <br/>";
            echo " Fichier: $fichier <br/>";
            echo " Ligne: $ligne <br/>";
                                    }, E_WARNING
                        );
   echo $a; // le handler ne sera pas exécuté
   // Le fichier inconnu.php étant pas existant
   include('inconnu.php');  // La fonction anonyme sera invoquée ici
   require('inconnu.php');
   echo  'hello' ;
?>
```

Les exceptions en PHP

- Depuis sa version 5, PHP propose une autre manière de gérer les erreurs que celle d'utiliser un gestionnaire d'erreur que nous passons à la méthode « **set_error_handler** »
- Dans cette nouvelle gestion, les erreurs sont appelées des «**exceptions**»
- Les exceptions sont des objets de la classe « **Exception** » générés (instanciées) lorsqu'une erreur se produit durant l'exécution du script
- Le principe de la gestion des exceptions est **d'attraper** (**catch**) l'exception une fois générée
- La gestion des exceptions est plus fluide et permet de personnaliser le comportement du programme en cas d'erreur

Lancer et attraper une exception

- Pour gérer des erreurs via des exceptions, il faut passer par les étapes suivantes :
  - Définir/déterminer quand une exception est (potentiellement) générée avec l'instruction **throw**
  - Utiliser un bloc **try** dans lequel mettre le code qui va potentiellement générer l'exception
  - Création d'un bloc **catch** dont le but est d'attraper l'exception et de définir le comportement du programme si l'erreur se produit
- Notez qu'il est possible de définir nos propre exceptions.Il suffit de créer la class de l'exception qui doit hériter « extends » de la classe « Exception » de PHP

Exemple

```php
include_once('DivisionByZeroException.php');

$a = 20;
$b = 0;
try {
    echo calculerDivision($a, $b);
} catch (DivisionByZeroException $dbze){
    echo json_encode(array(
                        "error" => array(
                            "message" => $dbze->getMessage(),
                            "code" => $dbze->getCode()
                            )
                        )
                    , JSON_UNESCAPED_UNICODE                    );
}

function calculerDivision(int $a, int $b){
    if($b == 0){
        throw new DivisionByZeroException("Divion par zéro non autorisé", 9004);
    }
    return $a / $b;
}
```

```php
<?php
    class DivisionByZeroException extends Exception {

    }
?>
```