Try, catch et finally

"Attraper" les erreurs

Pour "attraper" une erreur pour la gérer d'une certaine manière, il faut utiliser des blocs `try / catch / finally`.

Cela peut être utile dans de nombreux cas : par exemple, réessayer une requête, afficher un message spécifique etc.

Une erreur qui n'est pas attrapée sera transformée en une erreur fatale sauf si un gestionnaire global d'exception a été défini.

`Try`

Le bloc `try`, pour "essayer", permet d'exécuter une ou plusieurs instructions qui peuvent potentiellement échouer.

`Catch`

Le bloc `catch` permet d'exécuter une ou plusieurs instructions dans le cas où une exception a été lancée (`throw`).

Plusieurs blocs `catch` peuvent être définis pour gérer différentes classes d'exceptions.

`Finally`

Le bloc `finally` permet de toujours exécuter une ou plusieurs instructions, indépendamment du fait qu'une exception ait été lancée.

Autrement dit, les instructions présentes dans `finally` seront toujours exécutées.

Exemples

Prenons un exemple avec une fonction qui ne doit recevoir que des entiers positifs.

```php
<?php

error_reporting(E_ALL);
ini_set('display_errors', '1');

function accepteEntierPositif(int $arg)
{
  if ($arg < 0) {
    throw new UnexpectedValueException('Reçoit uniquement des entiers positifs');
  }
}


try {
  accepteEntierPositif(-10);
} catch (TypeError $e) {
  echo 'Exception TypeError reçue : ',  $e->getMessage(), "
";
} catch (UnexpectedValueException $e) {
  echo 'Exception UnexpectedValueException reçue : ',  $e->getMessage(), "
";
}
```

Ici nous irons dans le second `catch` car nous avons une exception `UnexpectedValueException`.

Réessayons avec une valeur qui n'est pas un entier :

```php
<?php

error_reporting(E_ALL);
ini_set('display_errors', '1');

function accepteEntierPositif(int $arg)
{
  if ($arg < 0) {
    throw new UnexpectedValueException('Reçoit uniquement des entiers positifs');
```

```
  }
}


try {
  accepteEntierPositif("bonjour");
} catch (TypeError $e) {
  echo 'Exception TypeError reçue : ',  $e->getMessage(), "
";
} catch (UnexpectedValueException $e) {
  echo 'Exception UnexpectedValueException reçue : ',  $e->getMessage(), "
";
}
```

Ici nous aurons dans le premier `catch` car l'erreur est une `TypeError`.

En ajoutant un bloc `finally`, nous pouvons exécuter des instructions mêmes s'il n'y a pas d'exception :

```
<?php

error_reporting(E_ALL);
ini_set('display_errors', '1');

function accepteEntierPositif(int $arg)
{
  if ($arg < 0) {
    throw new UnexpectedValueException('Reçoit uniquement des entiers positifs');
  }
}


try {
  accepteEntierPositif(42);
} catch (TypeError $e) {
  echo 'Exception TypeError reçue : ',  $e->getMessage(), "
";
} catch (UnexpectedValueException $e) {
  echo 'Exception UnexpectedValueException reçue : ',  $e->getMessage(), "
";
} finally {
  echo "Terminé !";
}
```

Ici nous afficherons `Terminé !`.

Les méthodes disponibles sur les `Exception`

La classe `Exception` dispose d'un ensemble d'accesseurs permettant de récupérer des éléments de l'exception :

```
getMessage();   // message de l'exception
getCode();  // code de l'exception
getFile();   // nom du fichier source
getLine();    // ligne du fichier source
getTrace();   // un tableau de backtrace()
getPrevious();   // exception précédente (depuis PHP 5.3)
getTraceAsString();  // chaîne formatée de trace
```