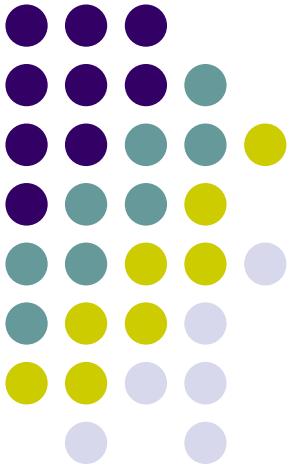
PHP

Exceptions



Franck LAMY – 2023/2024 – BTS SIO2

Objectifs

- **Introduire la notion d'exceptions**
- **Mettre en œuvre des exceptions**

Présentation

Les exceptions offrent une manière structurée et puissante de gérer les erreurs et les situations imprévues dans le code.

Présentation

Les exceptions permettent de séparer le code d'erreur du code normal, facilitant ainsi la lecture et la maintenance du code.

Plus de if...else pour gérer les erreurs

Exceptions

Définition

- Une **exception** est une **condition d'erreur** au moment de l'**exécution**

- Une **exception** est un **objet** et s'intègrent naturellement dans les principes et pratiques de la **POO**

Exceptions

Exemple

Retourne un bool

```php
class Validateur {
    public function verifieNombre(int $nombre): bool {
    // test si le nombre est strictement positif
        if ($nombre < 0) {
            return false;
        }
        return true;
     }
}
```

Code appelé

```php
$validateur = new Validateur();
if ($validateur->verifieNombre(10)) {
    echo "Le nombre est valide.";
} else {
    echo "Le nombre doit être positif.";
}
```

Code appelant

if...else

Exemple

```php
public function verifieNombre(int $nombre): bool
{
    // test si le nombre est un entier
    if ($nombre < 0) {
        throw new \Exception('Le nombre doit être positif.');
    }
    return true;
}
```

Code appelé

On lance une exception

throw

Exceptions

Lancer exception

Le message à afficher

La classe de l'exception

```php
throw new \Exception('Le nombre doit être positif.');
```

throw

Création d'une instance de la classe Exception

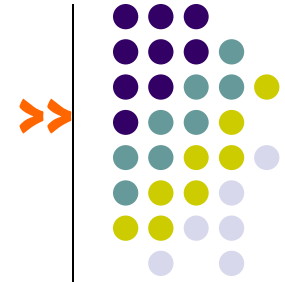
L'exception à lancer !

Lancer exception

- Créer d'une instance de l'exception : classe Exception
- Lancer l'exception : permet de signaler qu'une condition d'erreur s'est produite

throw

exception



Tests

Exceptions

Try...Catch

exception

Code appelé

Code appelant

throw

try...catch

Exceptions

Try...Catch

throw

exception

Code appelant

```php
$validateur = new Validateur();
try {
  $validateur->verifieNombre(10);
  echo "Le mot de passe est valide.";
}
```

Je surveille si une exception est lancée

Exceptions

Try...Catch

throw

exception


```php
$validateur = new Validateur();
try {
   $validateur->verifieNombre(10);
   echo "Le mot de passe est valide.";
} catch (\Exception $e) {
     echo $e->getMessage();
}
```

Code appelant

Je surveille si une exception est lancée

Exceptions

Try...Catch

Exception lancée

```php
$validateur = new Validateur();
try {
  $validateur->verifieNombre(-5);
  echo "Le mot de passe est valide.";
} catch (\Exception $e) {
    echo $e->getMessage();
}
```

Code appelant

Je surveille si une exception est lancée

L'exception interceptée (attrapée)

Hiérarchie

- En PHP, les **exceptions** sont **organisées** dans une **hiérarchie de classes**

- **Exception** : classe de base pour toutes les exceptions en PHP

Exception personnalisée

Possibilité de créer ses propres classes d'exception

Les noms des classes d'exception personnalisées permettant de décrire avec précision le type d'erreur ou la condition exceptionnelle qu'elles représentent, ce qui rend le code plus lisible et auto-documenté.

Exceptions

Exception personnalisée

```php
class NombreException extends \Exception
{}
```

```php
public function verifieNombre(int $nombre): bool
{
    // test si le nombre est un entier
    if ($nombre < 0) {

        throw new NombreException('Le nombre doit être
        positif.');

    }
    return true;
}
```

Code appelé

Exceptions

Exception personnalisée

```php
class NombreException extends \Exception
{}
```

```php
public function verifieNombre(int $nombre): bool
{
    // test si le nombre est un entier
    if ($nombre < 0) {

        throw new NombreException('Le nombre doit être
        positif.');

    }
    return true;
}
```

Code appelé

```php
$validateur = new Validateur();
try {
  $validateur->verifieNombre(10);
  echo "Le mot de passe est valide.";
} catch (\NombreException $e) {
    echo $e->getMessage();
}
```

Code appelant

Travaux pratiques

Ajouter dans la classe Validateur une méthode permettant de valider un mot de passe

Un mot de passe est valide si :

- La longueur est >= 8 caractères
- Contient au moins un chiffre
- Contient au moins une majuscule
- Contient au moins une minuscule