

POO ++

Michael
X  NATIS

1. L'opérateur **\$this**
2. Les **constructeurs**
3. Les **exceptions**
4. Les éléments **static**
5. Les **namespaces**

L'opérateur **\$this**

```
<?php

class User {

    private $nom;

    public function getNom(): string {
        return $this->nom;
    }

    public function setNom(string $value) {
        $this->nom = $value;
    }

}

$maxime = new User();
$maxime->setNom('Maxime');
echo ($maxime->getNom());
// Output : Maxime

?>
```

L'opérateur **\$this**
permet de désigner
l'objet courant

Les **setters** et les
getters permettent un
accès contrôlé aux
propriétés

Les **constructeurs**


```
<?php  
  
class User {  
    function __construct(string $nom, string $email) {  
    }  
}  
  
$maxime = new User('Maxime', 'Lemaire');  
  
?>
```

Les **constructeurs**
permettent de
paramétrer la
construction d'un objet.
C'est la 1^{ère} méthode
qui est appelée

Les exceptions

Lancement d'une erreur (exception)

```
<?php  
  
throw new Exception("L'utilisateur n'a pas d'adresse email");  
  
?>
```

Réaction à une exception

```
<?php

try {

    // Code avec une exception potentielle

} catch (Exception $e) {

    echo 'Une erreur est survenue: ' . $e->getMessage();

}

?>
```

Les **exceptions**
permettent une **gestion**
d'erreur facile mais
sont couteuses
techniquement

Les éléments **static**

```
1  <?php
2
3  class User {
4
5      public static $Prenom = 'Sandra';
6
7      public static function sePresenter() {
8          echo self::$Prenom;
9      }
10
11     public static function changerDePrenom() {
12         self::$Prenom = 'Jean';
13     }
14 }
15
16
17 $sandra = new User();
18 $sandra->sePresenter();
19
20 $jean = new User();
21 $jean->sePresenter();
22
23 $jean->changerDePrenom();
24 $jean->sePresenter();
25 $sandra->sePresenter();
26
27 // Utilisation des méthodes de classes
28 User::changerDePrenom();
29 User::sePresenter();
30
31 ?>
```

```
1  <?php
2
3  class User {
4
5      public static $Prenom = 'Sandra';
6
7      public static function sePresenter() {
8          echo self::$Prenom;
9      }
10
11     public static function changerDePrenom() {
12         self::$Prenom = 'Jean';
13     }
14
15 }
16
17 $sandra = new User();
18 $sandra->sePresenter();
19 // Output: Sandra
20
21 $jean = new User();
22 $jean->sePresenter();
23 // Output: Sandra
24
25 $jean->changerDePrenom();
26 $jean->sePresenter();
27 // Output: Jean
28 $sandra->sePresenter();
29 // Output: Jean
30
31 // Utilisation des méthodes de classes
32 User::changerDePrenom();
33 User::sePresenter();
34
35 ?>
```

Les éléments **static** (propre
à la classe et non à l'objet)

**Les membres `static` sont
appelés des membres de
classe (propriétés de classe
ou méthodes de classe)**

Les namespaces

```
<?php

namespace App\Models {

    class User {
        public $prenom;
    }

}

?>
```

```
<?php

include 'User.php';

use App\Models\User;

$thomas = new User();

?>
```


Les namespaces
permettent d'organiser
les classes en groupe

