Commutateur

La structure de `switch` remplit la même fonction qu'une série d'instructions `if` , mais peut effectuer le travail avec moins de lignes de code. La valeur à tester, telle que définie dans l'instruction `switch` , est comparée en termes d'égalité avec les valeurs de chacune des instructions de `case` jusqu'à ce qu'une correspondance soit trouvée et que le code dans ce bloc soit exécuté. Si aucune instruction de `case` correspondante n'est trouvée, le code dans le bloc `default` est exécuté, s'il existe.

Chaque bloc de code dans une instruction `case` ou `default` doit se terminer par l'instruction `break` . Cela arrête l'exécution de la structure du `switch` et continue l'exécution du code immédiatement après. Si l'instruction `break` est omise, le code de l'instruction `case` suivante est exécuté, *même s'il n'y a pas de correspondance* . Cela peut entraîner une exécution de code inattendue si la déclaration `break` est oubliée, mais peut également être utile lorsque plusieurs instructions de `case` doivent partager le même code.

```
switch ($colour) {
case "red":
   echo "the colour is red";
   break;
case "green":
case "blue":
   echo "the colour is green or blue";
   break;
case "yellow":
   echo "the colour is yellow";
   // note missing break, the next block will also be executed
case "black":
   echo "the colour is black";
   break;
default:
   echo "the colour is something else";
   break;
}
```

En plus de tester des valeurs fixes, la construction peut également être contraint de tester des déclarations dynamiques en fournissant une valeur booléenne au `switch` déclaration et toute expression au `case` déclaration. Gardez à l'esprit que la *première* valeur correspondante est utilisée, le code suivant affichera «plus de 100»:

```
$i = 1048;
switch (true) {
case ($i > 0):
   echo "more than 0";
```

```
    break;
case ($i > 100):
    echo "more than 100";
    break;
case ($i > 1000):
    echo "more than 1000";
    break;
}
```

10 façons de coder la même chose en PHP

On commence par se passer des if/else

Version 1 : données d'entrée et de sortie, un for, du if/else et plein de variables

```
// Données d'entrée
$input = 'PFCFFPC';

// Résultat attendu
$result = 'FCPCCFP';

$response = '';

$length = strlen($input);
for ($i = 0; $i < $length; $i++) {

        $play = $input[$i];

        if ($play === 'P') {
                $counterPlay = 'F';
        } elseif ($play === 'F') {
                $counterPlay = 'C';
        } else {
                $counterPlay = 'P';
        }

        $response .= $counterPlay;
}

// Test de la solution trouvée
if ($response === $result) {
        echo 'Challenge OK';
}
```

Il est important ici de bien effectuer le strlen avant le for. Sinon le strlen va s'effectuer à chaque itération.

Pour les prochains exemples, je ne reprendrais pas à chaque fois la déclaration de $input, $result, ni le test final.

Version 2 : j'enlève des variables et je rajoute un traitement d'erreur

```
$response = '';

$length = strlen($input);
for ($i = 0; $i < $length; $i++) {

        if ($input[$i] === 'P') {
                $response .= 'F';
        } elseif ($input[$i] === 'F') {
                $response .= 'C';
        } elseif ($input[$i] === 'C') {
                $response .= 'P';
        } else {
                // Erreur... encore un qui veut jouer Puit !
        }
}
```

Je teste donc bien précisément le cas du « C » et je garde un dernier else pour une éventuelle erreur.

Je fais l'économie de mes variables $play et $counterPlay.

Version 3 : je retire les else grâce au continue

```
$response = '';

$length = strlen($input);
for ($i = 0; $i < $length; $i++) {

        if ($input[$i] === 'P') {
                $response .= 'F';
                continue;
        }

        if ($input[$i] === 'F') {
                $response .= 'C';
                continue;
        }

        if ($input[$i] === 'C') {
                $response .= 'P';
                continue;
        }
```

```
            // Erreur... non, Puit ça n'existe pas !
}
```

Comme le continue va directement passer à l'itération suivante, plus besoin de else !

Version 4 : je retire les if grâce au switch

```
$response = '';

$length = strlen($input);
for ($i = 0; $i < $length; $i++) {

            switch ($input[$i]) {
                        case 'P':
                                    $response .= 'F';
                        break;

                        case 'F':
                                    $response .= 'C';
                        break;

                        case 'C':
                                    $response .= 'P';
                        break;

                        default:
                                    // Erreur... il triche le coquin
                        break;
            }
}
```

Le switch se prête très bien à ce genre de situation.

Version 5 : je passe du for au foreach avec str_split

```
$response = '';

foreach (str_split($input) as $play) {

            switch ($play) {
                        case 'P':
                                    $response .= 'F';
                        break;

                        case 'F':
                                    $response .= 'C';
                        break;
```

```
                case 'C':
                        $response .= 'P';
                break;

                default:
                        // Erreur... Euh... Lézard ? Sérieux ?
                break;
        }
}
```

Le str_split peut être mis directement dans le foreach, pas nécessaire de créer une variable avant.

Version 6 : je crée des constantes pour améliorer la lisibilité

```
define('PIERRE', 'P');
define('FEUILLE', 'F');
define('CISEAUX', 'C');

$response = '';

foreach (str_split($input) as $play) {

        switch ($play) {
                case PIERRE:
                        $response .= FEUILLE;
                break;

                case FEUILLE:
                        $response .= CISEAUX;
                break;

                case CISEAUX:
                        $response .= PIERRE;
                break;

                default:
                        // Erreur... Euh... Le Lézard mange la Feuille ?
                break;
        }
}
```