PHP : Conditions

Les **conditions** sont des blocs de code qui permettent d'exécuter certaines instructions en fonction de conditions spécifiques. En PHP, les structures conditionnelles comme `if`, `else`, et `elseif` sont essentielles pour contrôler le flux d'exécution d'un programme.

Qu'est-ce qu'une condition en PHP ?

Une condition est une instruction qui vérifie si une expression ou un ensemble d'expressions est **vrai** ou **faux**. En fonction du résultat, elle décide quelle partie du code sera exécutée.

La structure `if`

La structure `if` permet d'exécuter un bloc de code **seulement si** la condition spécifiée est évaluée à `true`.

Syntaxe :

```
if (condition) {
    // Code à exécuter si la condition est vraie
}
```

Exemple simple :

```
<?php
$age = 18;

if ($age >= 18) {
    echo "Tu es majeur.";
}
?>
```

Résultat :

```
Tu es majeur.
```

Dans cet exemple, le code à l'intérieur du bloc `if` est exécuté uniquement si `$age` est supérieur ou égal à 18.

La structure `else`

La structure `else` permet d'exécuter un autre bloc de code lorsque la condition dans le `if` est **fausse**.

Syntaxe :

```
if (condition) {
    // Code à exécuter si la condition est vraie
} else {
    // Code à exécuter si la condition est fausse
}
```

Exemple avec `else` :

```
<?php
$age = 16;

if ($age >= 18) {
    echo "Tu es majeur.";
} else {
    echo "Tu es mineur.";
}
?>
```

Résultat :

```
Tu es mineur.
```

La structure `elseif`

La structure `elseif` est utilisée pour vérifier plusieurs conditions successives. Si une condition est satisfaite, le bloc correspondant est exécuté, et les autres conditions sont ignorées.

Syntaxe :

```
if (condition1) {
    // Code si condition1 est vraie
} elseif (condition2) {
    // Code si condition2 est vraie
} else {
    // Code si aucune condition n'est vraie
}
```

Exemple avec `elseif` :

```
<?php
$note = 12;
if ($note < 10) {
    echo "Insuffisant.";
} elseif ($note < 15) {    // On sait d'office que $note >= 10
    echo "Passable.";
} else {
    echo "Très bien.";
}
?>
```

Résultat :
`Passable.`

Combiner `if`, `else`, et `elseif`

Tu peux utiliser plusieurs `elseif` pour gérer des scénarios complexes, suivis d'un `else` pour les cas par défaut.

Exemple : Afficher une évaluation selon l'âge

```php
<?php
$age = 25;

if ($age < 18) {
    echo "Mineur.";
} elseif ($age <= 25) {      // On sait d'office que $age >= 18
    echo "Jeune adulte.";
} elseif ($age <= 60) {      // On sait d'office que $age > 25
    echo "Adulte.";
} else {
    echo "Senior.";
}
?>
```

Résultat :
`Jeune adulte.`

Opérateurs courants dans les conditions

Les conditions utilisent des **opérateurs** pour comparer des valeurs ou vérifier des relations. Voici les principaux :

Opérateur	Description	Exemple
`==`	Égal à	`$a == $b`
`!=` ou `<>`	Différent de	`$a != $b` ou `$a <> $b`
`<`	Inférieur à	`$a < $b`
`>`	Supérieur à	`$a > $b`
`<=`	Inférieur ou égal à	`$a <= $b`
`>=`	Supérieur ou égal à	`$a >= $b`
`&&` ou `AND`	Et logique	`$a > 0 && $b > 0`
`` ` ``		`ou`OR``
`!`	Négation logique	`!$a` (faux si `$a` est vrai)

Exemples concrets

Exemple 1 : Vérification d'un login

```php
<?php
$username = "admin";
$password = "1234";

if ($username == "admin" && $password == "1234") {
    echo "Connexion réussie.";
} else {
    echo "Nom d'utilisateur ou mot de passe incorrect.";
}
?>
```

Résultat :

```
Connexion réussie.
```

Exemple 2 : Calcul des réductions

Un magasin applique des réductions en fonction du montant d'achat.

Code :

```php
<?php
$montant = 120;

if ($montant < 50) {
    echo "Pas de réduction.";
} elseif ($montant <= 100) { // On sait d'office que $montant >= 50
    echo "5% de réduction.";
} else {
    echo "10% de réduction.";
}
?>
```

Résultat :

```
10% de réduction.
```

Exemple 3 : Vérification du type de données

Utiliser `if` pour vérifier si une variable est du type attendu.

Code :

```php
<?php
$variable = 42;

if (is_int($variable)) {
    echo "C'est un entier.";
} elseif (is_string($variable)) {
    echo "C'est une chaîne.";
} else {
    echo "Type inconnu.";
}
?>
```

Résultat :

```
C'est un entier.
```

Exemple 4 : Conditions imbriquées

Les conditions peuvent être imbriquées pour gérer des scénarios complexes.

Code :

```php
<?php
$age = 20;
$carteEtudiant = true;

if ($age < 18) {
    echo "Entrée interdite.";
} else {
    if ($carteEtudiant) {
        echo "Entrée à tarif réduit.";
    } else {
        echo "Entrée au tarif plein.";
    }
}
?>
```

Résultat :

```
Entrée à tarif réduit.
```

Les pièges courants

1. **Confondre `=` et `==`** :
   - `=` est utilisé pour l'assignation, tandis que `==` est utilisé pour la comparaison.
   - Mauvais exemple :

     ```php
     if ($a = 5) {
         echo "Ceci sera toujours vrai !";
     }
     ```

   - Correct :

     ```php
     if ($a == 5) {
         echo "La condition est correcte.";
     }
     ```

2. **Oublier les parenthèses** :
   - Les parenthèses autour des conditions sont obligatoires :

     ```php
     if $a == 5 // Erreur
     if ($a == 5) // Correct
     ```

3. **Utiliser trop de `elseif`** :

- Pour des cas très complexes, utilise un `switch` pour simplifier le code.

Exercices pour t’entraîner

1. Crée une page PHP qui affiche "Bonjour" ou "Bonsoir" selon l'heure actuelle. Utilise la fonction `date("H")` pour récupérer l'heure.
2. Écris un script qui vérifie si un nombre est pair ou impair.
3. Crée une condition qui vérifie si une variable contient un texte ou un nombre et affiche un message différent pour chaque cas.

Conclusion

Les conditions `if`, `else`, et `elseif` sont fondamentales pour le développement en PHP. Elles te permettent de contrôler l'exécution du programme en fonction des données et des situations. Grâce aux nombreux exemples fournis, tu es maintenant prêt à écrire des scripts PHP dynamiques et intelligents. 🚀

N’oublie pas de pratiquer en expérimentant avec tes propres conditions !