

Cours découverte métier DevOps

Séquence 5 : Découverte de langage php

1. Comment signaler que ce qu'on écrit est en langage php ?

En fait, chaque partie en PHP doit être signalée au serveur. En effet les fichiers .php vont mélanger dans la joie et la bonne humeur les balises HTML avec son CSS et les instructions PHP. Il faut donc un moyen de signaler tout ça pour que le serveur sache quoi faire.

- L'instruction qui marque le début du PHP est : **<?php**
- L'instruction qui marque la fin du PHP est : **?>**
- Entre ces 2 marqueurs, tout est en PHP.

2. Pré-requis

Au préalable après avoir installé apache2, nous allons procéder à l'installation de php avec la commande suivante :

#apt-get install php

Après cette installation, une certaine configuration est nécessaire pour le bon fonctionnement. Pour cela, nous allons les paquets suivants : libapache2-mod-php comme suit :

#apt-get install libapache2-mod-php

3. L'instruction « echo »

Cette instruction permet d'afficher un texte à l'écran. Oui on commence par quelque chose de facile !

Soit la page PHP/Html suivante :

```
<?php
    echo "Ceci est du texte";
?>
```

Exemple :

Taper ce petit programme, nommez-le « **page.php** » et enregistrez-le dans votre répertoire de travail sur le serveur local.

```
<html>
    <head> </head>
    <body>
        <?php
            echo "Ceci est du texte";
        ?>
    </body>
</html>
```

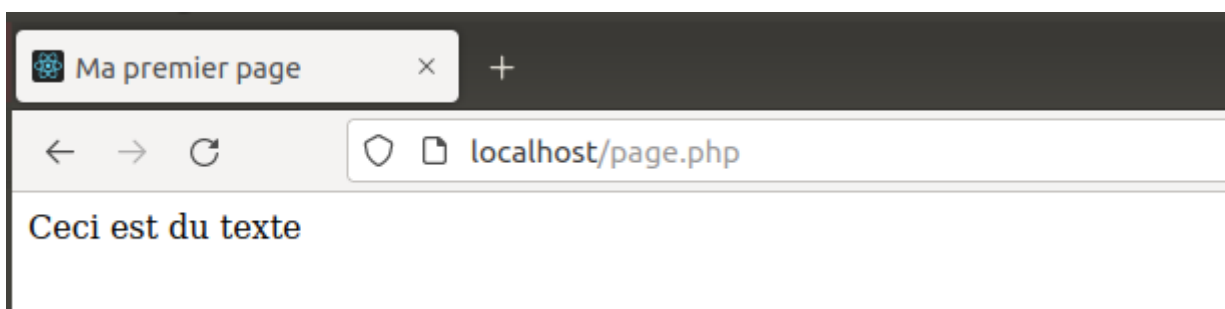
Avec l'éditeur vim, nous allons créer le fichier page.php dans /var/www/html.

```

<html>
    <head>
        <title> Ma premier page </title>
    </head>
    <body>
        <?php
            echo "Ceci est du texte";
        ?>
    </body>
</html>

```

Avec un navigateur, on peut accéder à la page comme suit :



4. Une variable en php

Comme PHP est un langage de programmation, on va pouvoir se servir de variables. Une variable étant une information stockée en mémoire, elle n'a pas une grande durée de vie. En PHP, la variable existe tant que la page est en cours de génération. Dès que la page PHP est générée, toutes les variables sont supprimées de la mémoire car elles ne servent plus à rien. Ce n'est donc pas un fichier qui reste stocké sur le disque dur mais une petite information temporaire présente en mémoire vive.

La variable est représentée par son nom précédé du \$ et elle peut être du type entier, float, chaîne de caractère,

exemples : \$nom= "toto" ; // variable de type chaîne de caractères
 \$taille = 12,5 ; // variable de type float (flottant)
 \$numero=3 ; // variable de type Int (entier)

Exemple :

Soit la page PHP/Html suivante:

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
  </head>
  <body>
    <h2>variables et Affichage d'un texte de la variable</h2>
    <p><br />
    <?php
        $age = 8;
        echo "Salut gamin !, je sais que tu as ";
        echo $age;
        echo " ans" ;

    ?>
  </p>
</body>
</html>
```

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
  </head>
  <body>
    <h2>variables et Affichage d'un texte de la variable</h2>
    <p><br />
    <?php
        $age = 8;
        echo "Salut gamin !, je sais que tu as ";
        echo $age;
        echo " ans";

    ?>
  </p>
</body>
</html>
```

Sur un navigateur :



5. Les méthodes de réception de données (get, post) dans un programme php

Faut-il utiliser plutôt la méthode GET pour envoyer les données d'un formulaire HTML, ou la méthode POST ?

GET et POST sont des méthodes d'accès définies dans le protocole HTTP et reprises dans la spécification HTML.

Le choix de la méthode dépend de la façon dont les données sont reçues, de la taille et de la nature des données.

- **La méthode GET ajoute les données à l'URL**

Dans un formulaire avec la balise **form** et **input** , elle est spécifiée ainsi :

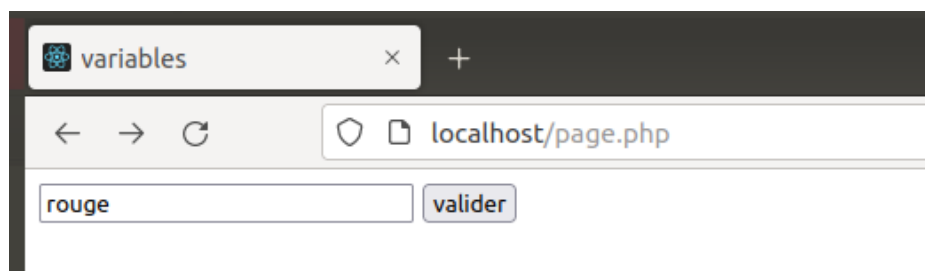
```
<form method="get" action="page.php">
  <input type="text" name='couleur' />
  <input type="submit" value="valider" />
</form>
```

Avec cette méthode, les données du formulaire seront encodées dans une URL. Celle-ci est composée du nom de la page ou du script à charger avec les données de formulaire empaquetées dans une chaîne.

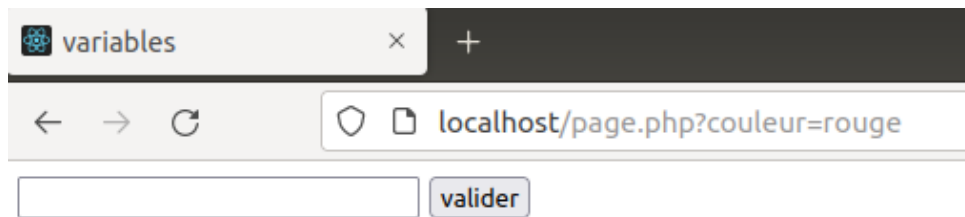
Les données sont séparées de l'adresse de la page par le code **?** et entre elles par le code **&**.

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
  </head>
  <body>
    <form method="get" action="page.php">
      <input type="text" name='couleur' />
      <input type="submit" value="valider" />
    </form>
  </body>
</html>
```

Sur un navigateur, on saisit la valeur dans la zone input:



Si on valide, on constate les données envoyées sur l'URL du navigateur:



- **La méthode POST n'a pas de taille limite**

Dans un fichier **page.php**, avec la balise **form** et **input** , elle est spécifiée ainsi:

```
<form method="post" action="page1.php">  
  <input type="text" name='couleur' />  
  <input type="submit" value="valider" />  
</form>
```

```
<!DOCTYPE html>  
<html xml:lang="fr" lang="fr">  
  <head>  
    <title>variables</title>  
  </head>  
  <body>  
    <form method="post" action="page1.php">  
      <input type="text" name='couleur' />  
      <input type="submit" value="valider" />  
    </form>  
  </body>  
</html>
```

Elle envoie un en-tête et un corps de message au serveur. Le corps est généralement constitué des données entrées dans le champ de formulaire par l'utilisateur.

Les données du formulaire n'apparaissent pas dans l'URL. En conséquence, il n'est pas possible de récupérer directement les données en JavaScript , il faut ajouter du code PHP dans la **page1.php**:

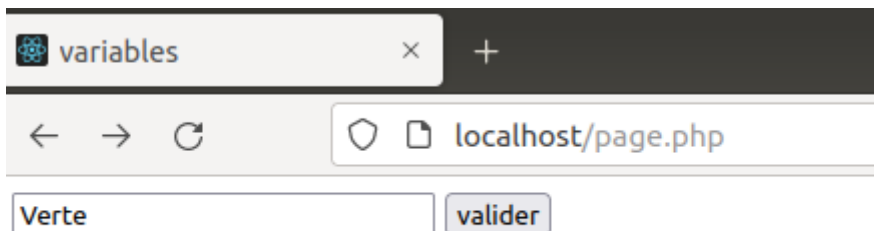
```
<?php  
    $couleur = $_POST['couleur'];  
    echo $couleur ;  
?>
```

```

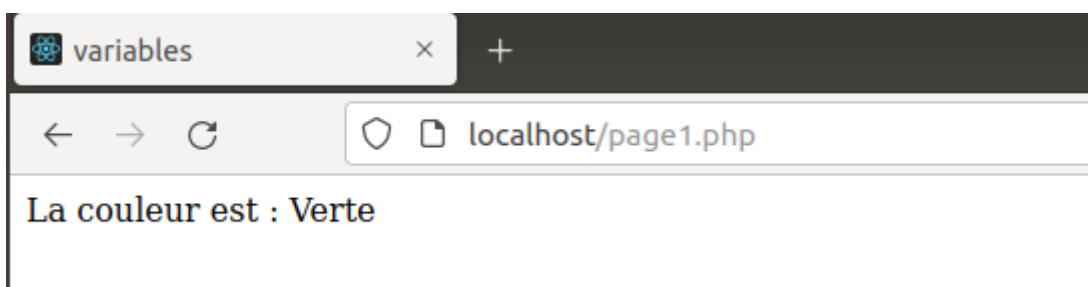
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
  </head>
  <body>
    <?php
      $couleur = $_POST["couleur"];
      echo 'La couleur est : '.$couleur ;
    ?>
  </body>
</html>

```

Dans un navigateur on exécute d'abord le fichier **page.php**, on remplit le formulaire et on valide.



Si on valide, on voit :



6. Envoi des données par la méthode get à partir d'un navigateur à un programme php

Il est possible d'envoyer des données dans une URL d'un navigateur et de les récupérer dans un autre programme php.

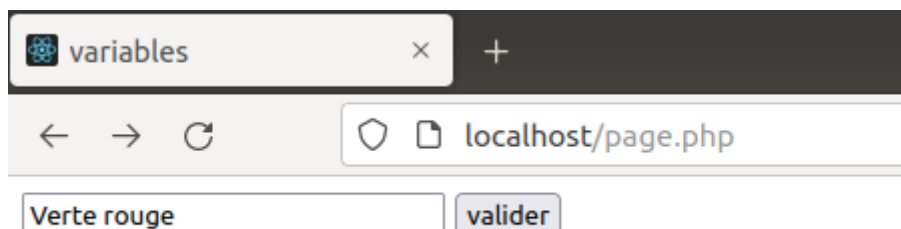
Dans le fichier **page.php**

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
  </head>
  <body>
    <form method="GET" action="page1.php">
      <input type="text" name="couleur"/>
      <input type="submit" value="valider"/>
    </form>
  </body>
</html>
```

Dans le fichier **page1.php**

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <?php
      $couleur = $_GET["couleur"];
      echo "La couleur à partir de l'URL est : ".$couleur;
    ?>
  </body>
</html>
```

Sur un navigateur, on exécute d'abord le fichier **page.php**, ensuite on est redirigé vers la **page1.php**.



On est redirigé vers la page1.php, on voit bien les données dans l'URL du navigateur et avec la méthode GET utilisée on parvient à traiter ces données.



7. Écriture d'un programme en php recevant un prénom et nom et qui affiche bonjour prénom nom

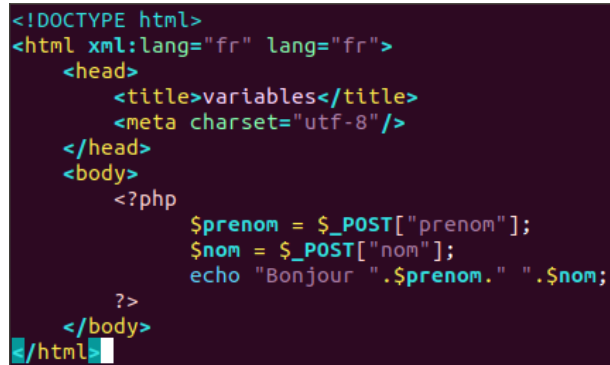
On crée le fichier **formulaire.html**, comme suit :

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <form method="POST" action="traitement.php">
      Prénom :<input type="text" name="prenom"/>
      Nom   :<input type="text" name="nom"/>
      <input type="submit" value="valider"/>
    </form>
  </body>
</html>
```

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <form method="POST" action="traitement.php">
      Prénom :<input type="text" name="prenom"/><br/>
      Nom   :<input type="text" name="nom"/><br/>
      <input type="submit" value="valider"/>
    </form>
  </body>
</html>
```

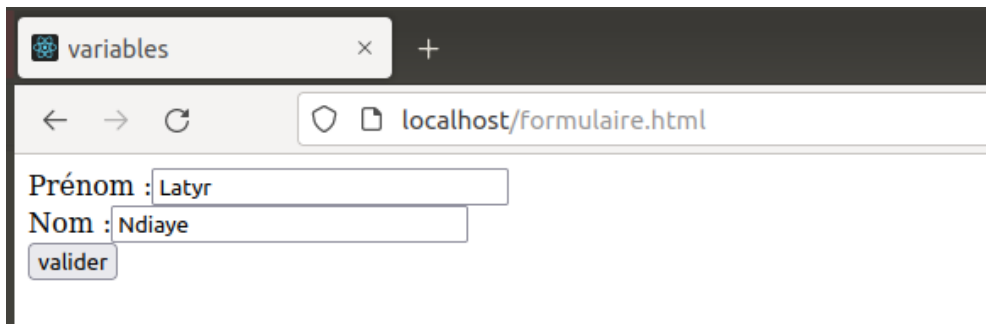

On crée le fichier **traitement.php** comme suit :

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <?php
      $prenom = $_POST["prenom"];
      $nom = $_POST["nom"];
      echo "Bonjour " . $prenom . " " . $nom;
    ?>
  </body>
</html>
```

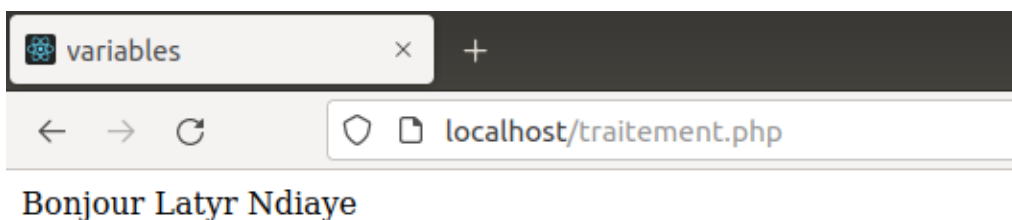
A screenshot of a code editor with a dark background. The code is the same as the one above, but with syntax highlighting: HTML tags are in blue, PHP tags are in green, and variables and strings are in yellow. The code is:

```
<!DOCTYPE html>
<html xml:lang="fr" lang="fr">
  <head>
    <title>variables</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <?php
      $prenom = $_POST["prenom"];
      $nom = $_POST["nom"];
      echo "Bonjour " . $prenom . " " . $nom;
    ?>
  </body>
</html>
```

Test sur le navigateur :

A screenshot of a web browser window. The title bar says 'variables'. The address bar shows 'localhost/formulaire.html'. The page content has two input fields: 'Prénom : Latyr' and 'Nom : Ndiaye'. Below the fields is a button labeled 'valider'.

Et si on valide on obtient :

A screenshot of a web browser window. The title bar says 'variables'. The address bar shows 'localhost/traitement.php'. The page content displays the text 'Bonjour Latyr Ndiaye'.