TD1 – Introduction à PHP Hello World, objets et formulaires

1. IDE pour PHP

PHP est un langage de programmation donc utilisez un environnement de développement. Vous ne codez pas du Java avec BlocNotes, c'est pareil pour PHP. Nous coderons donc notre PHP sous PhpStorm.

- Si vous êtes à l'IUT sous Linux, vous trouverez l'installation dans `/opt/phpstorm/`. Pour le lancer :

```
~/RepertoireCourant$  cd /opt/phpstorm/bin
/opt/phpstorm/bin$  ./phpstorm.sh
```

- Si vous utilisez votre propre machine, allez voir ces instructions dans les compléments du TDI.

i. Obtention de la licence académique Ultimate pour PhpStorm

- Normalement, vous avez obtenu une licence académique l'an dernier. Pour la retrouver, connectez-vous au site de JetBrains.
- Sinon, remplissez ce formulaire en utilisant votre adresse universitaire pour bénéficier d'une licence académique.

  Quelques minutes après, vous recevrez un email de confirmation suivi d'un second email d'activation où vous devrez accepter les conditions d'utilisation et choisir un nom d'utilisateur et un mot de passe. Conservez précieusement ces informations, car c'est grâce à elles que vous pourrez importer votre licence sur toutes les machines que vous allez utiliser (chez vous, à l'IUT etc).

ii. Documentations de PhpStorm

- Documentation officielle en anglais, dont notamment Step 1: Open a project in PhpStorm
- Documentation à l'IUT de Intellij Idea (proche de PhpStorm)

iii. Autre IDE

Si vous le souhaitez fortement, vous pouvez aussi utiliser d'autres IDE. VSCode est une bonne alternative, mais il manque des fonctionnalités PHP. Notez cependant que nous n'assurons pas le support d'autres IDE.

2. Accédez à vos pages web

Nous avons vu lors du cours 1 le fonctionnement du WWW sur le modèle de client & serveur HTTP. Mettons en pratique tout cela !

2.1 Une page HTML de base

Exercice 1

1. Créez une page **page1.html** avec le contenu suivant et enregistrez la dans le répertoire **public_html** de votre espace personnel.

```
<!DOCTYPE html>
<html>
    <head>
        <title> Insérer le titrer ici </title>
    </head>

    <body>
        Un problème avec les accents à é è ?
```

```
        <!-- ceci est un commentaire -->
    </body>
</html>
```

2. Ouvrez cette page dans le navigateur directement en double-cliquant dessus directement depuis votre gestionnaire de fichiers. Notez l'URL du fichier : file://chemin_de_mon_compte/public_html/page1.html.

   Que fait le gestionnaire de fichier quand on double-clique ?
   Que signifie le *file* au début de l'URL ?
   Est-ce que la page HTML s'affiche correctement ?
   Est-ce qu'il y a une communication entre un serveur et un client HTTP ?

   - **Rappel : Un problème avec les accents ?** Dans l'en-tête du fichier HTML vous devez rajouter la ligne qui spécifie l'encodage

     ```
     <meta charset="utf-8" />
     ```

     Il faut que vos fichiers soient enregistrés avec le même encodage. UTF-8 est souvent l'encodage par défaut, mais les éditeurs de texte offrent généralement le choix de l'encodage lors du premier enregistrement du fichier.

3. **Vous souvenez-vous comment fait-on pour qu'une page Web soit servie par le serveur HTTP de l'IUT (à l'URL http://webinfo.iutmontp.univ-montp2.fr/~mon_login_IUT/page1.html) ?**

   Réponse : Les pages Web doivent être enregistrées dans le dossier **public_html** de votre répertoire personnel.

   Ouvrez donc `page1.html` depuis le navigateur en tapant l'URL dans la barre d'adresse.

   Est-ce que la page HTML s'affiche correctement ?
   Est-ce qu'il y a une communication entre un serveur et un client HTTP maintenant ?

   Aide : Votre page ne s'affiche pas ?
   Si votre page ne s'affiche pas, c'est peut-être un problème de droit. Pour pouvoir servir vos pages, le serveur HTTP (Apache) de l'IUT doit avoir le droit de lecture des pages Web (permission `r--`) et le droit de traverser les dossiers menant à la page Web (permission `--x`). À l'IUT, la gestion des droits se fait par les ACL.
   Pour donner les droits à l'utilisateur www-data (Apache), utilisez la commande `setfacl` dans un terminal sous Linux :

   - `setfacl -m u:www-data:--x nom_du_répertoire` pour chaque répertoire menant à votre page Web.
   - Puis `setfacl -m u:www-data:r-- nom_de_la_page_Web`

   Note : Les ACL permettent d'avoir des droits spécifiques à plusieurs utilisateurs et à plusieurs groupes quand les droits classiques sont limités à un utilisateur et un groupe. Pour lire les droits ACL d'un fichier ou dossier, on tape `getfacl nom_du_fichier`.

2.2 Notre première page PHP

Exercice 2

1. Créez une page `echo.php` avec le contenu suivant.
   Pour ne pas que votre **public_html** devienne une décharge de pages Web à ciel ouvert, créez

des répertoires pour les cours et les TDs. Nous vous conseillons donc d'enregistrer `echo.php` dans `.../public_html/PHP/TD1/echo.php`.

```php
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title> Mon premier php </title>
    </head>

    <body>
        Voici le résultat du script PHP :
        <?php
          // Ceci est un commentaire PHP sur une ligne
          /* Ceci est le 2ème type de commentaire PHP
          sur plusieurs lignes */

          // On met la chaine de caractères "hello" dans la variable 'texte'
          // Les noms de variable commencent par $ en PHP
          $texte = "hello world !";

          // On écrit le contenu de la variable 'texte' dans la page Web
          echo $texte;
        ?>
    </body>
</html>
```

2. Ouvrez cette page dans le navigateur directement depuis votre gestionnaire de fichiers OU de façon équivalente avec une URL en `file://` comme : file://chemin_de_mon_compte/public_html/PHP/TDI/echo.php.

   Que se passe-t-il quand on ouvre un fichier PHP directement dans le navigateur ? Pourquoi ?
   Ça vous rappelle le cours I j'espère ?

3. Ouvrez cette page dans le navigateur dans un second onglet en passant par le serveur HTTP de l'IUT : http://webinfo.iutmontp.univ-montp2.fr/~mon_login/PHP/TDI/echo.php

   Que se passe-t-il quand on demande un fichier PHP à un serveur HTTP ? Regardez les sources de la page Web (Clic droit, code source ou `Ctrl-U`) pour voir ce qu'a vraiment généré PHP.
   N'hésitez pas à relire la partie du cours I concernée.

2.3 Notre premier dépôt Git

Ce cours de PHP est aussi l'occasion de manipuler le gestionnaire de version Git. Nous allons débuter en créant un *dépôt Git*, c'est-à-dire un dossier dans lequel la chronologie de toutes modifications pourront être enregistrées.

Exercice 3

1. Créons un dépôt Git sur GitLab (ou GitHub si vous y avez vos habitudes).
   i. Allez sur le GitLab de l'IUT et créez un nouveau projet `TD-PHP`.
   ii. Récupérez l'adresse de votre projet dans le bouton `Clone`.

   Note : Vous pouvez utiliser HTTPS ou SSH pour vous connecter. Nous vous recommandons SSH pour ne pas avoir à taper votre login/mot de passe à chaque `git push/pull`. Vous pouvez aller

voir les notes complémentaires au TDI pour recréer une clé SSH. Si vous utilisez HTTPS, vous aurez besoin d'exécuter d'abord la commande suivante pour que `git clone` marche.

```
# Pour anticiper une erreur due aux certificats de l'IUT
#               "server certificate verification failed"
git config --global http.sslverify false
```

   i. Dans le terminal, faites `git clone` suivi de l'adresse de votre projet.
   ii. Enfin déplacez le dossier `TD1` dans le répertoire `TD-PHP` créé par le `git clone`.

2. Nous allons configurer Git pour qu'il connaisse votre nom et votre adresse email (**étudiante**), ce qui sera utile quand vous travaillerez en groupe pour savoir qui a enregistré quelle modification :

```
git config --global user.name "Votre Prénom et Nom"
git config --global user.email "votreemail@umontpellier.fr"
```

   Aussi pour nous simplifier la vie plus tard, veuillez exécuter la commande suivante. Cela change l'éditeur de texte qu'ouvre Git par défaut.

```
git config --global core.editor "gedit --new-window -w"
```

3. Pour en savoir plus sur l'état de Git, **exécutez** la commande `git status`.

   La partie qui nous intéresse tout de suite est la suivante

```
Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)

  TD1/
```

   Elle nous dit que le suivi des modifications n'est pas activé pour le dossier `TD1`.

4. **Exécutez** la commande `git add TD1` pour suivre les modifications de tous les fichiers dans le répertoire `TD1`.

   Ré-exécutez la commande `git status` pour voir le changement :

```
Modifications qui seront validées :
  (utilisez "git rm --cached <fichier>..." pour désindexer)

    nouveau fichier : TD1/echo.php
```

5. Git a vu des modifications dans le fichier `TD1/echo.php` mais il reste encore à les enregistrer. Pour ceci, **exécutez** la commande `git commit` et écrivez un petit message de validation pour s'y retrouver plus tard (avant les lignes commentées avec #), puis fermez l'éditeur.

6. Une dernière exécution de `git status` nous renseigne que nous avons bien tout validé.

3. Les bases de PHP

3.1 Différences avec Java

1. Le code PHP doit être compris entre la balise ouvrante `<?php` et la balise fermante `?>`
2. Les variables sont précédées d'un `$`
3. Il n'est pas obligatoire de déclarer le type d'une variable (cf. plus loin dans le TD)

3.2 Les chaînes de caractères

Différentes syntaxes existent en PHP ; selon les délimiteurs que l'on utilise, le comportement est différent.

i. Avec guillements simples

Les chaînes de caractères avec ***simple quote*** **'** sont conservées telles quelles (pas de caractères spéciaux `\n`...). Les caractères protégés sont `'` et `\` qui doivent être échappés avec un anti-slash comme ceci `\'` et `\\`;

La concaténation de chaînes de caractères se fait avec l'opérateur point **.**

```php
$texte = 'hello' . 'World !';
```

ii. Avec guillements doubles

Pour simplifier le code suivant

```php
$prenom="Helmut";
echo 'Bonjour ' . $prenom . ', ça farte ?';
```

PHP propose une syntaxe de chaîne de caractères entourées de ***double quotes*** **"** qui permet d'écrire

```php
$prenom="Helmut";
echo "Bonjour $prenom, ça farte ?";
```

Les chaînes de caractères avec ***double quotes*** **"** peuvent contenir :

- des variables (qui seront remplacées par leur valeur),
- des sauts de lignes,
- des caractères spéciaux (tabulation `\t`, saut de ligne `\n`).
- Les caractères protégés sont `"`, `$` et `\` qui doivent être échappés avec un anti-slash `\` comme ceci : `\"`, `\$` et `\\`;

Astuce : En cas de problèmes avec le remplacement de variables, rajoutez des accolades autour de la variable à remplacer. Cela marche aussi bien pour les tableaux `"{$tab[0]}"`, les attributs `"{$objet->attribut}"` et les fonctions `"{$objet->fonction()}"`.

Documentation : Les chaînes de caractères sur PHP.net

Exercice 4

Qu'écrivent chacun des `echo` suivants ?

```php
$prenom = "Marc";

echo "Bonjour\n " . $prenom;
echo "Bonjour\n $prenom";
echo 'Bonjour\n $prenom';

echo $prenom;
echo "$prenom";
```

Testez votre réponse en rajoutant ce code dans `echo.php`.

Astuce:

- Vous pouvez aussi tester ce code dans le terminal (sans passer par un serveur Web et un navigateur). Pour cela, écrivez votre script dans `testEcho.php` (n'oubliez pas la balise ouvrante `<?php` en début de fichier). Puis, dans le terminal, exécutez `php testEcho.php`.

  Ce fonctionnement est plus proche de ce que vous auriez fait en Python avec `python script.py`, ou en Java avec `javac program.java` puis `java program` .

3.3 Affichage pour le débogage

Les fonctions `print_r` et `var_dump` affichent les informations d'une variable et sont très utiles pour déboguer notamment les tableaux ou les objets.
La différence est que `print_r` est plus lisible car `var_dump` affiche plus de choses (les types).

3.4 Les tableaux associatifs

Les tableaux en PHP peuvent aussi s'indexer par des entiers ou des chaînes de caractères :

- Pour initialiser un tableau, on utilise la syntaxe

```php
$utilisateur = array(
 'prenom' => 'Juste',
 'nom'    => 'Leblanc'
);
```

ou la syntaxe raccourcie équivalente

```php
$utilisateur = [
 'prenom' => 'Juste',
 'nom'    => 'Leblanc'
];
```

- On peut ajouter des cases à un tableau :

```php
$utilisateur['passion'] = 'maquettes en allumettes';
```

Note : Le tableau `$utilisateur` contient plusieurs associations. Par exemple, il associe à la chaîne de caractères `'prenom'` la chaîne de caractères `'Juste'`.
Dans cette association, `'prenom'` s'appelle la **clé** (ou **index**) et `'Juste'` la **valeur**.

- On peut rajouter facilement un élément "à la fin" d'un tableau avec

```php
$utilisateur[] = "Nouvelle valeur";
```

- Notez l'existence des boucles `foreach` pour parcourir les paires clé/valeur des tableaux.

```php
foreach ($mon_tableau as $cle => $valeur){
    //commandes
}
```

La boucle `foreach` va boucler sur les associations du tableau. Pour chaque association, `foreach` va mettre la clé de l'association dans la variable `$cle` et la valeur dans `$valeur` puis exécuter les commandes. Par exemple

```php
foreach ($utilisateur as $cle => $valeur){
    echo "$cle : $valeur\n";
}
```

va afficher ceci (ou l'inverse car l'ordre des entrées n'est pas assuré)

```
prenom : Juste
nom : Leblanc
passion : maquettes en allumettes
0 : Nouvelle valeur
```

Remarque : La boucle `foreach` est indispensable pour parcourir les indices et valeurs d'un tableau indexé par des chaînes de caractères.
Il existe aussi bien sûr une boucle `for` classique si le tableau est indexé uniquement par des entiers

```php
for ($i = 0; $i < count($mon_tableau); $i++) {
    echo $mon_tableau[$i];
}
```

Pour comprendre `foreach` autrement, le code suivant

```php
foreach ($mon_tableau as $cle => $valeur){
    //commandes
}
```

est équivalent à

```php
for ($i = 0; $i < count(array_keys($mon_tableau)); $i++) {
    $cle = array_keys($mon_tableau)[$i];
    $valeur = $mon_tableau[$cle];
    //commandes
}
```

Source : Les tableaux sur php.net

3.5 Exercices d'application

Exercice 5

1. Dans votre fichier `echo.php`, créez trois variables `$marque`, `$couleur` et `$immatriculation` contenant des chaînes de caractères de votre choix, et une variable `$nbSieges` contenant un nombre ;
2. Créez la commande PHP qui écrit dans votre fichier le code HTML suivant (en remplaçant bien sûr la marque par le contenu de la variable `$marque` ...) :

```html
<p> Voiture 256AB34 de marque Renault (couleur bleu, 5 sieges) </p>
```

3. Faisons maintenant la même chose mais avec un tableau associatif `voiture`:
    - Créez un tableau `$voiture` contenant quatre clés `"marque"`, `"couleur"`, `"immatriculation"` et `"nbSieges"` et les valeurs de votre choix ;
    - Utilisez l'un des affichages de débogage (*e.g.* `var_dump`) pour vérifier que vous avez bien rempli votre tableau ;
    - Affichez le contenu de la "voiture-tableau" au même format HTML

```html
<p> Voiture 256AB34 de marque Renault (couleur bleu, 5 sieges) </p>
```

4. Maintenant nous souhaitons afficher une liste de voitures :
    - Créez une liste (un tableau indexé par des entiers) `$voitures` de quelques "voitures-tableaux" ;
    - Utilisez l'un des affichages de débogage (*e.g.* `var_dump`) pour vérifier que vous avez bien rempli `$voitures` ;
    - Modifier votre code d'affichage pour écrire proprement en HTML un titre "Liste des voitures :" puis une liste (`<ul><li>...</li></ul>`) contenant les informations des voitures.
    - Rajoutez un cas par défaut qui affiche "Il n'y a aucune voiture." si la liste est vide.
      (On vous laisse chercher sur internet la fonction qui teste si un tableau est vide)
5. Enregistrez votre travail dans Git :
    i. Faites `git status` pour connaître l'état du dépôt Git.
    ii. Faites `git add TD1/echo.php` pour lui dire d'enregistrer les modifications dans `echo.php`.
    iii. Faites `git commit` pour valider l'enregistrement des modifications et écrivez un petit message de validation (comme e.g. *"TDI Exo4 Affichage de variables"*).

iv. Finissez par `git status` pour voir que tout s'est bien passé.

4. La programmation objet en PHP

PHP était initialement conçu comme un langage de script, mais est passé Objet à partir de la version 5. Plutôt que d'utiliser un tableau, créons une classe pour nos voitures.

4.1 Un exemple de classe PHP

Exercice 6

1. Créer un fichier **Voiture.php** avec le contenu suivant

```php
<?php
class Voiture {

    private $marque;
    private $couleur;
    private $immatriculation;
    private $nbSieges; // Nombre de places assises

    // un getter
    public function getMarque() {
        return $this->marque;
    }

    // un setter
    public function setMarque($marque) {
        $this->marque = $marque;
    }

    // un constructeur
    public function __construct(
      $marque,
      $couleur,
      $immatriculation,
      $nbSieges
   ) {
        $this->marque = $marque;
        $this->couleur = $couleur;
        $this->immatriculation = $immatriculation;
        $this->nbSieges = $nbSieges;
    }

    // une methode d'affichage.
    public function afficher() {
      // A compléter dans le prochain exercice
    }
}
?>
```

Notez les **différences avec Java** :

- Pour accéder à un attribut ou une fonction d'un objet, on utilise le `->` au lieu du `.` de Java.
- En PHP, `$this` est obligatoire pour accéder aux attributs et méthodes d'un objet.
- On doit mettre le mot-clé `function` avant de déclarer une méthode
- Le constructeur ne porte pas le nom de la classe, mais s'appelle `__construct()`.
- En PHP, on ne peut pas avoir deux fonctions avec le même nom, même si elles ont un

nombre d'arguments différent. En particulier, il ne peut y avoir au maximum qu'un constructeur.

2. Créez des *getter* et des *setter* pour `$couleur`, `$immatriculation` et `$nbSieges` ;
(PhpStorm peut les générer automatiquement pour vous avec Clic droit > Generate)

3. L'intérêt des *setter* est notamment de vérifier ce qui va être écrit dans l'attribut. Comme l'immatriculation est limitée à 8 caractères, codez un *setter* qui ne stocke que les 8 premiers caractères de l'immatriculation dans l'objet. Mettez aussi à jour le constructeur pour ne garder que les 8 premiers caractères.
(Documentation PHP : fonction `substr` *substring*).

4. Remplissez `afficher()` qui permet d'afficher les informations de la voiture courante.

5. Testez que votre classe est valide pour PHP : la page générée par le serveur Web `webinfo` à partir de `Voiture.php` ne doit pas afficher d'erreur.
Demandez donc votre page à `webinfo` http://webinfo.iutmontp.univ-montp2.fr/~mon_login/PHP/TDI/Voiture.php.

6. Enregistrez votre travail à l'aide de `git add` et `git commit`. Aidez-vous toujours de `git status` pour savoir où vous en êtes.

4.2 Utilisation de la classe `Voiture`

Nous voulons utiliser la classe `Voiture` dans un autre script `testVoiture.php`. Il faut donc inclure le fichier `Voiture.php`, qui contient la déclaration de la classe de `Voiture`, dans `testVoiture.php` pour pouvoir l'utiliser.

i. Require

PHP a plusieurs façons d'inclure un fichier :

- `require "dossier/fichier.php"` : inclut et exécute le fichier spécifié en argument, ici `"dossier/fichier.php"`. Autrement dit, tout se passe comme si le contenu de `fichier.php` avait copié/collé à la place du `require`.
Renvoie une erreur si le fichier n'existe pas.

- `require_once "dossier/fichier.php"` : fait de même que `require` mais la différence est que si le code a déjà été inclus, il ne le sera pas une seconde fois.
Ceci est particulièrement utile pour inclure une classe car cela assure qu'on ne l'inclura pas deux fois.

Notez qu'il existe `include` et `include_once` qui ont le même effet mais n'émettent qu'un warning si le fichier n'est pas trouvé (au lieu d'une erreur).

ii. Exercice

Exercice *7*

1. Créez un fichier **testVoiture.php** contenant le squelette HTML classique (`<html>`,`<head>`, `<body>` ...)

2. Dans la balise `<body>`, on va vouloir créer des objets `Voiture` et les afficher :
    - Incluez `Voiture.php` à l'aide de `require_once` comme vu précédemment ;
    - Initialisez une variable `$voiture1` de la classe `Voiture` avec la même syntaxe qu'en Java ;
    - Affichez cette voiture à l'aide de sa méthode `afficher()` .
3. Testez votre page sur `webinfo` :

http://webinfo.iutmontp.univ-montp2.fr/~mon_login/PHP/TDI/testVoiture.php

4.3 Déclaration de type

Exercice 8

Optionnellement, on peut déclarer les types de certaines variables PHP :

- les arguments d'une fonction
- la valeur de retour d'une fonction
- les attributs de classe

Ces types sont vérifiés à l'exécution, contrairement à Java qui les vérifie à la compilation.
La déclaration de type est **cruciale** pour que l'**autocomplétion** de l'IDE (PHPStorm, VSCode, ...) marche.

Exemple:

```php
class Requete {
  // Déclaration de type d'un attribut
  string $url;
  string $methode; // GET ou POST
}
class Reponse {
  int $code; // 200 OK ou 404 Not Found
  string $corps; // <html>...
}

// Déclaration de type d'un paramètre de fonction (Requete)
// et d'un retour de fonction (Reponse)
function ServeurWeb(Requete $requete) : Reponse (
// ...
)
```

Documentation PHP

1. Mettez à jour `Voiture.php` pour déclarer `marque`, `couleur` et `immatriculation` comme `string`, et `nbSieges` comme `int` dans
   - les attributs de classes
   - les arguments des setters
   - les sorties des getters
   - les arguments du constructeur
2. Testez que PHP vérifie bien les types : dans `testVoiture.php`, appelez une fonction qui attend en argument un `string` en lui donnant à la place un tableau (le tableau vide `[]` par exemple). Vous devez recevoir un message comme suit

   ```
   PHP Fatal error:  Uncaught TypeError: Voiture::__construct(): Argument #1
   ($marque) must be of type string, array given
   ```

3. Enregistrez votre travail à l'aide de `git add` et `git commit`. Aidez-vous toujours de `git status` pour savoir où vous en êtes.

5. Interaction avec un formulaire

Exercice 9

1. Créez un fichier **formulaireVoiture.html**, réutilisiez l'entête du fichier **echo.php** et dans le

body, insérez le formulaire suivant:

```
<form method="get" action="creerVoiture.php">
  <fieldset>
    <legend>Mon formulaire :</legend>
    <p>
      <label for="immat_id">Immatriculation</label> :
      <input type="text" placeholder="256AB34" name="immatriculation"
id="immat_id" required/>
    </p>
    <p>
      <input type="submit" value="Envoyer" />
    </p>
  </fieldset>
</form>
```

2. Souvenez-vous (ou relisez le cours 1) de la signification des attributs `action` de `<form>` et `name` de `<input>`

   Rappels supplémentaires :

   - L'attribut `for` de `<label>` doit contenir l'identifiant d'un champ `<input>` pour que un clic sur le texte du `<label>` vous amène directement dans ce champ.
   - l'attribut `placeholder` de `<input>` sert à écrire une valeur par défaut pour aider l'utilisateur.

3. Créez des champs dans le formulaire pour pouvoir rentrer la marque, la couleur et le nombre de sièges (`<input type="number">`) de la voiture.

4. Souvenez-vous (ou relisez le cours 1) de ce que fait un clic sur le bouton "**Envoyer**".
   Vérifiez en cliquant sur ce bouton.

   Comment sont transmises les informations ?
   Comment s'appelle la partie de l'URL contenant les informations ?

5. Prenez l'habitude d'enregistrer régulièrement votre travail sous Git. Vous pouvez utiliser la commande `git log` pour voir l'ensemble de vos enregistrements passés.

Exercice 10

1. Créez un fichier `creerVoiture.php` :
   i. Aidez-vous si nécessaire du cours 1 pour savoir comment récupérer l'information envoyée par le formulaire.
   ii. Vérifiez que `creerVoiture.php` reçoit bien des informations dans le *query string*. Pour cela, vérifiez que le tableau `$_GET` n'est pas vide.
   iii. En reprenant du code de `testVoiture.php`, faites que `creerVoiture.php` affiche les informations de la voiture envoyée par le formulaire.
   iv. Bien sûr, **testez votre page** en la demandant à `webinfo`.
2. Afin d'éviter que les données du formulaire n'apparaissent dans l'URL, modifiez le formulaire pour qu'il appelle la méthode POST :

   ```
   <form method="post" action="creerVoiture.php">
   ```

   et côté `creerVoiture.php`, mettez à jour la récupération des paramètres :

   i. Souvenez-vous (ou relisez le cours 1) de par où passe l'information envoyée par un formulaire de méthode POST ;

ii. Changez `creerVoiture.php` pour récupérer l'information envoyée par le formulaire ;
iii. Essayez de **retrouver l'information envoyée par le formulaire** avec les outils de développement (Onglet Réseau).

3. Avez-vous pensé à enregistrer vos modifications sous Git ? Faites le notamment en fin de TD pour retrouver plus facilement où vous en êtes la prochaine fois.
   Note : Vous pouvez faire `git diff` à tout moment pour voir les modifications que vous avez faites depuis la dernière validation (`commit`).

6. Les bases d'un site de covoiturage

Exercice II

Vous allez programmer les classes d'un site de covoiturage, dont voici la description d'une version minimaliste:

- **Utilisateur :** Un utilisateur possède des champs propres `(login, nom, prenom)`
- **Trajet :** Une annonce de trajet comprend :
  1. un identifiant unique `id`,
  2. les détails d'un trajet (un point de départ `depart` et un point d'arrivée `arrivee`),
  3. des détails spécifiques à l'annonce comme une date de départ `date`,
  4. un nombre de places disponibles `nbplaces`,
  5. un prix `prix`,
  6. et le login du conducteur `conducteur_login`,

7. Installez un serveur Apache chez vous

Nous vous conseillons d'installer Apache + PhP + MySql + PhpMyAdmin sur votre machine perso, ce sera très utile pour la suite et notamment pour le projet.

Installation :

- sous Linux : Au choix
  - XAMP
    https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql/preparer-son-environnement-de-travail#/id/r-4443743
  - LAMP
    https://doc.ubuntu-fr.org/lamp
    Vérifiez que vous installez bien aussi `phpmyadmin` et que vous activez le module Apache `userdir` pour pouvoir mettre vos pages Web dans `public_html`.
- sous Mac OS X & Windows (MAMP) :
  https://openclassrooms.com/fr/courses/918836-concevez-votre-site-web-avec-php-et-mysql/4237816-preparez-votre-environnement-de-travail#/id/r-4443661

Attention, pensez à modifier le `php.ini` pour mettre `display_errors = On` et `error_reporting = E_ALL`, pour avoir les messages d'erreurs. Car par défaut, le serveur est configuré en mode production (`display_errors = Off`). Il faut redémarrer Apache pour que les modifications soient prises en compte.
