

LABORATOIRE 3

CRÉATION DE FICHIERS LIBRAIRIES

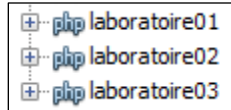

INSERTION DE CODE *PHP* DANS UNE PAGE *XHTML*

1. Présentation

Les deux principaux objectifs de ce laboratoire sont d'illustrer comment (1) créer et utiliser un fichier librairie et (2) insérer du code *PHP* dans une page *XHTML*.

2. Pour commencer;

Création d'un nouveau projet à partir d'un projet existant

1. En *Windows*, effectuez une copie du dossier `L:\420-4W5\laboratoire02` que vous nommerez `L:\420-4W5\laboratoire03`.
2. Ouvrez le dossier **laboratoire03**, puis **nbproject**. Notez la présence du dossier *private* et des fichiers *project.properties* et *project.xml*.
3. En utilisant l'éditeur *Notepad++*, ouvrez le fichier **project.xml**.
 - a. Remplacez l'occurrence de *laboratoire02* pour **laboratoire03**.
 - b. Enregistrez la modification, puis fermez le fichier.
4. Ouvrez le dossier **private**, puis en utilisant l'éditeur *Notepad++*, ouvrez le fichier **private.properties**.
 - a. Remplacez les deux occurrences de *laboratoire02* pour **laboratoire03**.
 - b. Enregistrez les modifications, puis fermez le fichier.
5. Démarrez l'application **NetBeans IDE**.
6. Cliquez sur **File/Open Project**, puis modifiez le dossier courant pour `L:\420-4W5`. Remarquez la présence des dossiers *php laboratoire01*, *php laboratoire02* et *php laboratoire03*.
 
7. Double-cliquez sur le nom **laboratoire03**. Le nouveau projet s'affiche dans l'arborescence.
8. Assurez-vous que le nom de projet sélectionné est bien **laboratoire03**, puis cliquez sur le bouton **RUN PROJECT**  de la barre d'outils *Run* pour déclencher le processus de téléversement. La boîte de dialogue *File Upload* s'affiche.
9. Assurez-vous que la case **index.php** est cochée, puis cliquez sur le bouton **UPLOAD**. Le téléversement s'effectue, puis après quelques secondes l'application s'exécute. Une sortie identique au laboratoire 2 s'affiche dans la fenêtre de navigateur.

Nous pouvons commencer !

4. Premier problème : Création de deux fichiers librairie *PHP*

a. Rappel : Qu'est-ce qu'un fichier librairie ?

Un fichier librairie est un fichier pouvant contenir des variables et des fonctions qui pourront servir à une ou plusieurs applications.

b. Rappel : Types de fichier librairie

Il existe essentiellement deux types de fichiers librairie :

1. Librairie utilitaire : Les variables et fonctions qui y sont entreposées pourront être utilisées par plus d'une application.
Ex. : *librairie-generale.php*
2. Librairie spécifique : Les variables et fonctions qui y sont entreposées ne pourront être utilisées que pour une application en particulier.
Ex. : *librairie-projet01.php*

c. Stratégies à adopter pour l'entreposage des variables et fonctions utilitaires

Deux stratégies peuvent être adoptées soit...

1. Création d'un seul fichier librairie contenant toutes les variables et fonctions (Ex. : *librairie.php*)
2. Création de plusieurs fichiers librairie contenant les variables et fonctions regroupées selon leur champ d'action. (Ex. : *librairie-bd.php*, *librairie-date.php*, etc.).
Il est suggéré de regrouper tous les fichiers librairie utilitaires dans un seul dossier pour en faciliter leur gestion.

Quelle que soit la stratégie adoptée pour l'entreposage de vos fonctions, assurez-vous qu'elles ne contiennent aucune erreur de syntaxe ou de logique !

d. Création du fichier librairie *librairie-date.php*

1. Cliquez droit sur le nom de projet **laboratoire03** (ou sur le dossier **Source Files** du projet *laboratoire03*) pour afficher son menu contextuel, puis sur **New/PHP File....**
2. Remplacez le nom actuel (*newEmptyPHP*) pour « **librairie-date** ». Ne saisissez pas le suffixe **.php**, car il sera inséré automatiquement (voir la zone *Created File*).
3. Cliquez sur **FINISH**.
4. Remplacez le contenu actuel du fichier pour...

```
<?php
?>
```

5. Ouvrez le fichier **index.php** (du projet *laboratoire03*), sélectionnez tout le code source de la fonction **extraitJJMMAAAA** (incluant la boîte de commentaire), puis déplacez-le entre les balises **<?php** et **?>** du fichier librairie.
6. Effectuez la même manœuvre pour le code source de la fonction **moisEnLitteral**. Assurez-vous de déplacer le code après celui de la fonction *extraitJJMMAAAA*. **Pour faciliter le repérage, il est souhaitable d'entreposer les fonctions en ordre alphabétique.**
7. Déplacez le code source de la fonction **jourSemaineEnLitteral** entre les fonctions *extraitJJMMAAAA* et *moisEnLitteral*.
8. Pour terminer, déplacez le code source de la fonction **extraitJSJJMMAAAA** entre les fonctions *extraitJJMMAAAA* et *jourSemaineEnLitteral*.

e. Exécution de l'application *index.php*

1. Exécutez l'application sur le serveur. (Assurez-vous d'avoir téléversé les fichiers !)

Un problème se pose : Deux messages d'erreur s'affichent !

! Fatal error: Uncaught Error : Call to undefined function *extraitJJMMAAAA()* in C:\wamp\www\nomutilisateur\laboratoire03\index.php on line *nn*

! Error: Call to undefined function *extraitJJMMAAAA()* in C:\wamp64\www\nomutilisateur\laboratoire03\index.php on line *nn*

Call Stack

#	Time	Memory	Function	Location
1	0.0000	999999	{main}()	..\index.php:0

Remarquez que les messages d'erreur indiquent que la fonction *extraitJJMMAAAA()* n'est pas définie.

Notez que c'est normal étant donné que nous avons déplacé le code source de la fonction dans le fichier *librairie-date.php* !

2. Pour corriger la situation, nous devons inclure le code source du fichier **librairie-date.php** dans l'application *index.php*.

Ajoutez la ligne ci-dessous comme première ligne de code de l'application *index.php*.

```
<?php

    require_once "librairie-date.php";

    $strDate = date("d-m-Y");
    ...
```

Réf. : <http://php.net/manual/fr/function.require-once.php>

3. Enregistrez la modification apportée, puis exécutez l'application sur le serveur pour constater que tout fonctionne correctement.

f. Exercice 1 : Création du fichier librairie *librairie-generale.php*

1. Créez le fichier librairie **librairie-generale.php** dans le projet *laboratoire03*.
2. Déplacez le code source des fonctions **convertitSousChaineEnEntier** et **er** de l'application *index.php* vers le nouveau fichier librairie.
3. Saisissez l'instruction **require_once** qui permettra d'inclure ladite librairie dans le fichier *index.php*.
4. Exécutez l'application sur le serveur. Tout fonctionne correctement.

5. Deuxième problème :

Affichage de la date courante dans une page XHTML

On vous demande de créer une nouvelle application *index.php* qui permet d'afficher la date courante dans une page XHTML.

Réf. : <http://424w.cgodin.qc.ca/lmbrousseau/laboratoire03/probleme2.php>

a. Pour commencer : un peu de ménage

1. Fermez tous les fichiers ouverts dans l'éditeur de texte.
2. Renommez le fichier **index.php** du projet *laboratoire03* pour **index-laboratoire02.php**.
3. Créez un nouveau fichier **index.php**.

```
<?php
?>
```

b. Création du fichier *index.css*

1. Créez le fichier **index.css** (*Cascading Style Sheet*) dans le projet *laboratoire03*.

2. Remplacez le code source existant pour le code ci-dessous...

```
BODY { font-family:arial; font-size:16px; padding-left:15px; padding-top:15px; }
TABLE { border-collapse: collapse; }
TD { border:solid 1px black; }

.sTitreApplication { font-size:32px; line-height:26px; font-weight:bold; margin-top:0px; }
.sTitreSection { font-size:24px; line-height:18px; font-weight:bold; }

.sDroits { font-size:12px; }
.sGras { font-weight:bold; }
.sRouge { color:red; }
```

3. Enregistrez les modifications, puis fermez le fichier.

c. Création du fichier d'inclusion *en-tete.php*

Nous allons maintenant créer un fichier d'inclusion que nous pourrons insérer au début de chacune des applications *PHP* que nous créerons à partir de maintenant.

1. Créez le fichier **en-tete.php**, puis effacez son contenu.
2. Entrez le code source ci-dessous :

```
<!DOCTYPE html>
<html>
<head>
  <title>Titre de l'application</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <link rel="stylesheet" type="text/css" href="index.css" />
</head>
<body>
  <form id="frmSaisie" method="" action="">
    <div id="divEntete" class="">
      <p class="sTitreApplication">
        Titre de l'application
        <span class="sTitreSection">
          <br />par <span class="sRouge">Nom de l'auteur</span>
        </span>
      </p>
    </div>
```

3. Comme vous pouvez l'observer, le fichier ne contient pas une once de code *PHP*. Pour le moment, du moins.
4. Enregistrez les modifications, puis fermez le fichier.

d. Création du fichier d'inclusion *pied-page.php*

Nous allons maintenant créer un fichier d'inclusion que nous pourrons insérer à la fin de chacune des applications *PHP* que nous créerons à partir de maintenant.

1. Créez le fichier **pied-page.php**, puis effacez son contenu.

- Entrez le code source ci-dessous :

```
<div id="divPiedPage">
  <p class="sDroits">
    &copy; Département d'informatique G.-G.
  </p>
</div>
</form>
</body>
</html>
```

- Comme vous pouvez l'observer, toujours aucune trace de code *PHP*.

- Enregistrez les modifications, puis fermez le fichier.

e. Génération d'une page Web à partir des deux fichiers d'inclusion

Nous allons maintenant générer une page Web à partir des deux fichiers d'inclusion créés précédemment.

- Saisissez le contenu ci-dessous dans le fichier *index.php*...

```
<?php
  require_once "en-tete.php";
  require_once "pied-page.php";
?>
```

- Exécutez l'application sur le serveur.
- Affichez maintenant le code source de ladite page. Intéressant, non ?

Titre de l'application
par **Nom de l'auteur**

© Département d'informatique G.-G.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Titre de l'application</title>
5   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6   <link rel="stylesheet" type="text/css" href="index.css" />
7 </head>
8 <body>
9   <form id="frmSaisie" method="" action="">
10    <div id="divEntete" class="">
11      <p class="sTitreApplication">
12        Titre de l'application
13        <span class="sTitreSection">
14          <br />par <span class="sRouge">Nom de l'auteur</span>
15        </span>
16      </p>
17    </div>
18    <div id="divPiedPage">
19      <p class="sDroits">
20        &copy; Département d'informatique G.-G.
21      </p>
22    </div>
23  </form>
24 </body>
25 </html>
```

f. Insertion de code source *PHP* dans du code source *XHTML*1. Ouvrez le fichier **en-tete.php**, puis apportez les modifications suivantes :

a. Remplacez...

```
<title>Titre de l'application</title>
```

Pour...

```
<title><?php echo $strTitreApplication; ?></title>
```

b. Remplacez...

```
href="index.css"
```

Pour...

```
href="<?php echo $strNomFichierCSS; ?>"
```

c. Remplacez...

```
<p class="sTitreApplication">
  Titre de l'application
```

Remarquez la présence
du « \n ».

Pour...

```
<p class="sTitreApplication">
  <?php echo "$strTitreApplication\n"; ?>
```

d. Remplacez...

```
<span class="sRouge">Nom de l'auteur</span>
```

Pour...

```
<span class="sRouge"><?php echo $strNomAuteur; ?></span>
```

2. Exécuter l'application *index.php* à ce stade serait problématique parce que les variables ***\$strTitreApplication***, ***\$strNomFichierCSS*** et ***\$strNomAuteur*** n'ont pas été initialisées. Pour vous en convaincre, exécutez l'application sur le serveur maintenant.3. Corrigez la situation en entrant les valeurs ci-dessous dans le fichier *index.php* pour chaque variable utilisée dans le fichier *en-tete.php* ...

```
<?php
  $strTitreApplication = "Affichage de la date courante";
  $strNomFichierCSS = "index.css";
  $strNomAuteur = "Entrez votre nom complet ici";
  require_once "en-tete.php";
  require_once "pied-page.php";
?>
```

4. Exécutez l'application sur le serveur.

Affichage de la date courante par Louis-Marie Brousseau

© Département d'informatique G.-G.

g. Affichage de la date courante

Nous sommes enfin prêts à afficher la date courante dans le corps de la page (entre les balises `<body></body>`).

Première étape : Inclusion des fichiers librairies

Puisque nous aurons besoin des fonctions de librairie entreposées dans les fichiers *librairie-date.php* et *librairie-generale.php*, nous allons inclure les deux fichiers comme premières lignes du code source :

```
<?php
    require_once "librairie-date.php";
    require_once "librairie-generale.php";

    $strTitreApplication = "Affichage de la date courante";
```

Deuxième étape : Création du corps de la page Web qui affichera la date courante

Comme vous l'avez déjà constaté...

- Le produit de l'exécution d'une application *PHP* apparaîtra du côté client sous forme de code *XHTML/CSS/JavaScript*.
- Il n'existe aucune façon pour l'utilisateur de voir le code source *PHP* à partir du navigateur.
- Le code *PHP* doit être inclus entre les balises `<?php` et `?>` pour être traité comme tel.
- Tout code qui n'est pas entre les balises ci-dessus mentionnées est ignoré par l'interpréteur *PHP* et est envoyé directement au navigateur.

Puisque la balise `<body>` a déjà été insérée à la fin du fichier d'inclusion *en-tete.php* et que la balise `</body>` l'a été au début du fichier d'inclusion *pied-page.php*, nous sommes prêts à écrire directement le corps de la page.

1. Modifiez le contenu de l'application *index.php* comme suit...

```
...
require_once "en-tete.php" ;
?>
<div id="divCorps" class="">
    C'est ici que nous afficherons la date courante
</div>
<?php
    require_once "pied-page.php";
?>
```

2. Exécutez l'application sur le serveur.

Affichage de la date courante

par **Louis-Marie Brousseau**

C'est ici que nous afficherons la date courante

© Département d'informatique G.-G.

Quelques remarques :

Comme indiqué précédemment, tout code source qui n'est pas entre les balises `<?php` et `?>` est envoyé directement au navigateur.

C'est ce qui explique que nous avons dû fermer la première balise `<?php` pour pouvoir insérer notre code *XHTML* et en ouvrir une deuxième pour que la dernière instruction (`require_once`) puisse être interprétée de façon PHP-esque.

3. Nous aurions pu également tout écrire en *PHP*. Pour vous en convaincre, effectuez les modifications suivantes, puis exécutez l'application sur le serveur.

```
require_once "en-tete.php";

echo "<div id=\"divCorps\" class=\"\">\n";
echo "    C'est ici que nous afficherons la date courante (approche PHP pur)\n";
echo "</div>\n";

require_once "pied-page.php";
```

Notez cependant que cette approche ne devrait pas être encouragée, car elle pourrait se révéler lourde si la partie *XHTML* est volumineuse. (Ce qui est la plupart du temps le cas !)

4. Annulez les dernières modifications apportées au code source et exécutez de nouveau l'application sur le serveur pour vous assurer que tout est correct.
5. Saisissons maintenant le minimum de code *PHP* permettant d'afficher la date courante. (Aux fins de la démonstration, nous supposons que la date courante est le 1^{er} février 2018.)

```
<?php
    require_once "librairie-date.php";
    require_once "librairie-generale.php";

    $strTitreApplication = "Affichage de la date courante";

    $strNomFichierCSS = "index.css";
    $strNomAuteur = "Entrez votre nom complet ici";
    require_once "en-tete.php";

    /* Déclaration des variables nécessaires au traitement */
    $intJourSemaine;
    $intJour;
    $intMois;
    $intAnnee;
    $strDate = "01-02-2018";
```

```

/* Extraction du jour de la semaine, du jour, du mois et de l'année
 * de la date courante. */
extraitJSJMMMAAAA($intJourSemaine, $intJour, $intMois, $intAnnee, $strDate);
?>
<div id="divCorps" class="">
  <!-- Affichage de la date courante en utilisant l'approche mixte XHTML/PHP -->
  Nous sommes le
  <?php echo jourSemaineEnLitteral($intJourSemaine); ?>
  <?php echo er($intJour); ?>
  <?php echo moisEnLitteral($intMois); ?>
  <?php echo $intAnnee; ?>.
  <br /><br />
</div>
<?php
  require_once "pied-page.php";
?>

```

6. Exécutez l'application sur le serveur.
7. Assurez-vous de bien comprendre la solution proposée avant de poursuivre.
8. Effectuez une copie du fichier *index.php* que vous nommerez **probleme2.php**.

Affichage de la date courante par **Louis-Marie Brousseau**

Nous sommes le jeudi 1^{er} février 2018.

© Département d'informatique G.-G.

6. Troisième problème :
Affichage de la date complète du premier samedi de chaque mois de l'année 2018

L'association *Les Amis de l'Informatique* vous demande de lui écrire une application qui permet d'afficher la date complète du premier samedi de chaque mois de l'année 2018.

Avant de résoudre ce problème, vous devrez concevoir deux fonctions utilitaires que vous copierez éventuellement dans leur librairie respective.

Réf. : <http://424w.cgodin.qc.ca/lmbrousseau/laboratoire03/probleme3.php>

Dates de réunion par **Louis-Marie Brousseau**

Réunion no 01 : **6 janvier 2018**

Réunion no 02 : **3 février 2018**

Réunion no 03 : **3 mars 2018**

Réunion no 04 : **7 avril 2018**

Réunion no 05 : **5 mai 2018**

Réunion no 06 : **2 juin 2018**

Réunion no 07 : **7 juillet 2018**

Réunion no 08 : **4 août 2018**

Réunion no 09 : **1^{er} septembre 2018**

Réunion no 10 : **6 octobre 2018**

Réunion no 11 : **3 novembre 2018**

Réunion no 12 : **1^{er} décembre 2018**

© Département d'informatique G.-G.

a. Exercice 2 : Création de la fonction *ajouteZeros*

- Après avoir consulté la documentation sur la fonction PHP `sprintf()`, on vous demande de concevoir la fonction **ajouteZeros** qui permet de préfixer un nombre d'une série de zéros.

Réf. : <http://php.net/manual/fr/function.sprintf.php>
<http://php.net/manual/fr/function.die.php>
<http://php.net/manual/fr/function.exit.php>

Prototype de la fonction à concevoir :

function ajouteZeros(\$numValeur, \$intLargeur)

Exemple d'appel :

```
for ($i=0; $i<=10; $i++) {
    echo ajouteZeros(1, $i) . "<br />";
}
die();
```

Sortie attendue :

```
1
1
01
001
0001
00001
000001
0000001
00000001
000000001
0000000001
```

- Lorsque la fonction sera validée, déplacez là dans le fichier de librairie **librairie-generale.php**.

Solution proposée :

```
/*
-----|
ajouteZeros (2018-mm-jj)
Scénario : ajouteZeros($numValeur, $intLargeur)
-----|
*/
function ajouteZeros($numValeur, $intLargeur) {
    $strFormat = "%0" . $intLargeur . "d";
    return sprintf($strFormat, $numValeur);
}

for ($i=0; $i<=10; $i++) {
    echo ajouteZeros(1, $i) . "<br />";
}
die();
```

b. Exercice 3 : Création de la fonction *JJMMAAAA*

- On vous demande de concevoir la fonction **JJMMAAAA** qui permet de retourner n'importe quelle date avec le format *jj-mm-aaaa* à partir de trois entiers (*Jour, Mois et Année*). La date n'a pas à être validée et pour ajouter un peu de piquant, l'année peut être saisie sous forme de 2 ou 4 chiffres.

Si la date est saisie sous forme de 2 chiffres alors...

0 à 20 ⇒ 2000 à 2020 et
21 à 99 ⇒ 1921 à 1999

Prototype de la fonction à concevoir :

function JJMMAAAA(\$intJour, \$intMois, \$intAnnee)

Exemple d'appel :

```
for ($i=0; $i<=99; $i++) {
    echo JJMMAAAA(1, 1, $i) . "<br />";
}
echo "<br />";
echo JJMMAAAA(1, 1, 2018) . "<br />";
echo JJMMAAAA(31, 12, 2018);
die();
```

2. Lorsque la fonction sera validée, déplacez là dans le fichier de librairie **librairie-date.php**.

Sortie attendue :

01-01-2000	01-01-1921	01-01-1981
01-01-2001	01-01-1922	01-01-1982
01-01-2002	01-01-1923	01-01-1983
01-01-2003	01-01-1924	01-01-1984
01-01-2004	01-01-1925	01-01-1985
01-01-2005	01-01-1926	01-01-1986
01-01-2006	01-01-1927	01-01-1987
01-01-2007	01-01-1928	01-01-1988
01-01-2008	01-01-1929	01-01-1989
01-01-2009	01-01-1930	01-01-1990
01-01-2010	01-01-1931	01-01-1991
01-01-2011	01-01-1932	01-01-1992
01-01-2012	01-01-1933	01-01-1993
01-01-2013	01-01-1934	01-01-1994
01-01-2014	01-01-1935	01-01-1995
01-01-2015	01-01-1936	01-01-1996
01-01-2016	01-01-1937	01-01-1997
01-01-2017	01-01-1938	01-01-1998
01-01-2018	01-01-1939	01-01-1999
01-01-2019	01-01-1940	
01-01-2020	...	01-01-2018
...	...	31-12-2018

Solution proposée :

```
/*
|-----|
| JJMMAAAA (2018-mm-jj)
| Scénario : JJMMAAAA($intJour, $intMois, $intAnnee) = > "$intJour-$intMois-$intAnnee"
| Si $intAnnee sur 2 positions
| Si $intAnnee <= 20 => 2000 à 2020 autrement => 1921 à 1999
|-----|
*/
function JJMMAAAA($intJour, $intMois, $intAnnee) {
    $intAnnee = $intAnnee <= 20 ? 2000 + $intAnnee :
                ($intAnnee <= 99 ? 1900 + $intAnnee : $intAnnee);
    /* La date retournée doit avoir le format 0J-0M-AAAA */
    return ajouteZeros($intJour, 2) . "-" .
           ajouteZeros($intMois, 2) . "-" .
           ajouteZeros($intAnnee, 4);
}

for ($i=0; $i<=99; $i++) {
    echo JJMMAAAA(1, 1, $i) . "<br />";
}
echo "<br />";
echo JJMMAAAA(1, 1, 2018) . "<br />";
echo JJMMAAAA(31, 12, 2018);
die();
```

c. Exercice 4 : Résoudre le problème maintenant !

Voici une piste de solution, car contrairement aux exercices précédents, aucune solution proposée n'est offerte (même si ce n'est pas totalement vrai) :

Pour chacun des douze mois de l'année 2018...

1. Entrez la date correspondant au premier jour du mois dans **\$strDate** en utilisant la fonction **JJMMAAAA()**.
2. En utilisant les fonctions **date()** et **strtotime()**, récupérez le jour de la semaine correspondant à **\$strDate**, puis entreposez-le dans **\$intJourSemaine**.
3. En fonction de **\$intJourSemaine**, assignez à **\$intPremierSamediMois** un nombre qui doit être dans l'intervalle 1 à 7.
4. En utilisant les fonctions **ajouteZeros()**, **er()** et **moisEnLitteral()**, affichez la date formatée.
5. N'oubliez pas de changer le contenu de la variable **\$strTitreApplication** pour « **Dates de réunion** ».

7. Pour terminer

1. Effectuez une copie du fichier *index.php* que vous nommerez **probleme3.php**.
2. Compressez le dossier **laboratoire03**, puis envoyez-le via LÉA. Avisez le professeur que vous avez terminé.

FIN DU LABORATOIRE...