PHP 06 - Les dates et les heures

Un développeur travaille en permanence avec des dates :

- date d'une facture client
- date inscription d'un membre sur un site
- calcul d'un délai de livraison, c'est-à-dire d'une différence nette deux dates (intervalle)
- ajout d'un nombre de jours ou de mois à une date

PHP fournit un ensemble d'outils pour les manipuler.

Les bases

Date locale (timezone)

Les dates et surtout les heures pour une localisation donnée sont variables selon le fuseau horaire. Il est donc essentiel de paramétrer le serveur d'hébergement en fonction de la situation géographique de vos internautes. Cette notion se nomme **timezone**.

En effet, si vous consultez en France un site canadien (par exemple) hébergé sur un serveur situé au au Canada, l'heure sera celle du Canada.

Par défaut, Wamp est configuré sur l'heure universelle GMT (méridien de Greenwich), il y aura donc un décalage (de 1 à 2 heures selon l'heure d'hiver/été) avec la France.

Pour obtenir la bonne heure, il faut configurer la valeur de l'option `datetime_zone` sur la valeur *Europe/Paris*. 2 solutions possibles :

- la plus simple : ajouter l'instruction `date_default_timezone_set("Europe/Paris");` dans vos scripts avant toute manipulation de dates. Dans ce cas, le fuseau horaire ne sera configuré que pour votre application.
- plus complexe : configurer cette valeur directement dans php.ini (voir avec un formateur); dans ce cas la modification sera effective pour l'ensemble des projets sur votre serveur Wamp.

La notion de timestamp

Le PHP gère les dates sous la forme d'un entier représentant le nombre de secondes écoulées depuis le **1er janvier 1970 0h00 00s GMT**, date issue de l'apparition d'Unix.

Ce nombre est appelé **timestamp** (= timbre de temps).

Ainsi, dans un script PHP, quand quelque chose est daté en timestamp, il s'agit du nombre de secondes écoulées depuis le 01/01/1970 à laquelle l'élément du script est créé.

On récupère le timestamp grâce à la fonction PHP `time()` :

```
<?php
echo time();
```

Le code ci-dessus affichera, par exemple, `1562225602` correspondant au 04/07/2019 09h32 22 secondes et quelques secondes.

Pour calculer une durée, il faudra alors récupérer le timestamp de fin (par exemple maintenant) et le déduire du timestamp de début, le résultat sera un nombre de secondes qu'il faudra convertir (en mois, en jours ou heures...).

Il faut donc pouvoir récupérer une date à partir d'un timestamp ou à l'inverse créer un timestamp pour une date donnée.

> Un timestamp peut être stocké dans une base de données avec le type de colonne *timestamp*.

Date du jour

En PHP, la fonction de base pour manipuler les dates est `date()`, qui s'utilise comme ceci :

```
<?php
$date = date("d/m/Y");
echo"Nous sommes le ".$date;

// Ou directement :
echo"Nous sommes le ".date("d/m/Y");
```

Ici, la fonction `date()` retourne la date du jour sous la forme suivante *12/11/2018* correspondant au format (un pattern) indiqué en paramètre :

- `d` : le jour, sur 2 chiffres,
- `m` : le mois, sur 2 chiffres,
- `Y` : l'année, sur 4 chiffres.
- Les slashes (`/`) servent uniquement à la présentation (séparateurs). On pourrait les remplacer par un tiret, un point ou deux points par exemple.

> Il existe de nombreux formatages possibles, attention à la casse (la lettre `d` en minuscule fournit un résultat différent que `D` en majuscule).

Heure

Pour obtenir l'heure, on utilise également la fonction `date()` en lui passant des paramètres de formatage spécifiques :

```
<?php
echo date("H:i:s");
```

Cet exemple affichera :

- `H` : heure, au format 24 avec zéros.
- `i` : minutes, avec zéros,
- `s` : secondes, avec zéros.

L'objet `DateTime`

Il est désormais recommandé de manipuler les dates en mode objet grâce à l'objet natif *DateTime*. Cette classe propose un ensemble de méthodes (fonctions) pour faciliter la manipulation (formatage, calculs, intervalles) des dates. Les versions récentes de PHP ont supprimé des fonctions de date traditionnelles ce qui oblige désormais à passer par *DateTime* pour certaines manipulations.

> Nous aborderons la programmation orientée objet dans une prochaine séquence.

La syntaxe est la suivante :

```
<?php
// on déclare une instance de l'objet PHP 'DateTime' :
$oDate = new DateTime();
```

Ce code retourne automatiquement la date et heure de l'instant où il est exécuté, c'est-à-dire celle du jour donc. Si on exécute le code suivant (affichage du contenu de la variable `$oDate`) :

```
<?php
var_dump($oDate);
```

Affiche :

```
C:\wamp\www\php_session\index.php:13:
2
object(DateTime)[1]
  public 'date' => string '2018-07-03 07:12:27.043832' (length=26)
  public 'timezone_type' => int 3
  public 'timezone' => string 'UTC' (length=3)
```

Pour travailler sur une autre date que celle du jour, par exemple une valeur de date enregistrée en base de données, il faut passer la valeur en argument :

```
<?php
$oDate = new DateTime("01-01-2018");
```

Formater une date existante

On a aussi souvent besoin d'afficher une date autre que celle du jour, par exemple une date qui provient de la base de données et qui donc est dans un format MySql :

- date, par exemple *2018-11-16*
- datetime, par exemple *2018-11-16 11:26:51*

Pour formater cette date, on va utiliser là encore l'objet PHP `DateTime` :

```
<?php
// $mecolonne_sql est issue d'une requête SQL, avec pour valeur 2018-11-16 11:26:51 (par exemple)
$oDate = new DateTime($mecolonne_sql);
echo $oDate->format("d/m/Y H:\h1");
```

Ceci affichera `16/11/2018 11h26` (l'affichage des secondes a été volontairement omis).

Pour en savoir plus.

Calcul sur les dates

Pour effectuer des calculs sur les dates,

> Attention, ces fonctions peuvent provoquer des bugs.

Vérifier une date

Essayez ce code de validation d'une date par une expression régulière :

```
$datePattern = "/^([0-3][0-9])-([0-1][0-9])-([0-3][0-9])$/";
$date = "01/10/2019";

if (preg_match($datePattern, $date)
{
    echo"Date ".$date." valide.<br>";
}
else
{
    echo"Date ".$date." erronée.<br>";
}
```

Jusque là tout va bien, la date est valide.

Essayez maintenant avec la date *32/13/2019* : la regex indique qu'elle est valide ! En effet, une expression régulière ne vérifie que le format, par les valeurs.

Pour s'assurer que les valeurs de jour, mois et année sont correctes, PHP propose la fonction `checkdate()` :

```
    $date = "01/10/2019";

    /*
    * - On découpe la chaine $date selon '/' avec la fonction explode() qui retourne un tableau
    * - Les éléments du tableau sont directement affectés à des variables ($dd, $mm, $yyyy) dans un ordre respectif grâce à la fonction list()
    */
    list($dd, $mm, $yyyy) = explode('/', $date);

    /* Les variables $dd, $mm, $yyyy sont des chaines, or la fonction checkdate()
    * n'accepte que des entiers
    */
    echo gettype($dd); // indique que $dd est une chaine

    /* On va donc 'caster' c'est-à-dire changer le type d'une variable,
    * ici on veut un entier, la syntaxe est de l'indiquer avec (int)
    * devant la variable
    */
    $dd = (int) $dd;

    echo gettype($dd); // $dd est désormais bien un entier

    $mm = (int) $mm;
    $yyyy = (int) $yyyy;

    if (!checkdate($mm, $dd, $yyyy))
    {
        echo"Date ".$date." est erronée.";
    }
```

Récupérer les erreurs de l'objet `DateTime`

L'objet `DateTime` permet de récupérer les erreurs grâce à `DateTime::getlastErrors` :

```
$dateTime = DateTime::createFromFormat('m/d/Y', $date);

$errors = DateTime::getlastErrors();

if (!empty($errors['warning_count']))
{
    return FALSE;
}
```

Documentation

- Liste des fonctions pour les dates et les heures
- L'objet *DateTime* (et les autres objets de dates : *DateInterval*, *DateTimeZone* etc.)

Exercices

Utilisez l'objet *DateTime*, sauf mention contraire.

Exercice 1

Affichez la date du jour au format *mardi 2 juillet 2019*.

Exercice 2

Trouvez le numéro de semaine de la date suivante : *14/07/2019*.

Exercice 3

Combien reste-t-il de jours avant la fin de votre formation.

Exercice 4

Reprenez l'exercice 3, mais traitez le problème avec les fonctions de gestion du timestamp, `time()` et `mktime()`.

Exercice 5

Quelle sera la prochaine année bissextile ?

Exercice 6

Montrez que la date du 17/17/2019 est erronée.

Exercice 7

Affichez l'heure courante sous cette forme : *11h25*.

Exercice 8

Ajoutez 1 mois à la date courante.