DateTime, la mejor forma para trabajar con fechas de cualquier formato en PHP

Hoy en día, rara es la aplicación web que no tenga que hacer uso de fechas para trabajar. El ejemplo más claro lo tenemos en cualquier blog o periódico digital donde al lado de cada noticia aparece su fecha de publicación, y en algunos casos incluso la hora. Aunque trabajar con este tipo de información no resulta complicado, para muchos se convierte en un auténtico dolor de cabeza debido a los distintos formatos que nos podemos encontrar. Hoy, en nuestro White Paper, os explicaremos el que quizás sea el método más sencillo para crear y convertir fechas en PHP, uno de los lenguajes de programación más utilizado para el desarrollo de páginas web.

Objeto DateTime de PHP

<? php date ...

El principal problema que nos podemos encontrar a la hora de trabajar con fechas son los muchos formatos que hay. Por ejemplo, no se representa igual la fecha cuando se hace en formato inglés a cuando lo hacemos en formato español. Otro de los casos más habituales es cuando se guardan en base de datos, ya que se suelen almacenar de una forma que no suele ser muy útil a la hora de mostrarla en la web.

Para que trabajar con fechas no sea tan problemático y podamos darle el formato que mejor se adapte a nuestras necesidades en cada momento, PHP nos ofrece un objeto muy útil para ello. Nos referimos al objeto DateTime.

Creando el objeto DateTime partiendo de cualquier cadena origen

index.php

```php
<?php
$cadenaFecha = "2017-01-09";

$fecha = DateTime:: createFromFormat('Y-m-d',$cadenaFecha);

$fecha2 = date_create_from_format('Y-m-d',$cadenaFecha);
```

En este apartado, lo que veremos será lo sencillo que resulta crear uno de estos objetos. Una vez que lo tengamos creado convertirlo en otro formato resultará una tarea muy sencilla.

Para llevar a cabo este proceso partiremos de una fecha base que bien podría haber sido sacada de una base de datos de nuestro alojamiento web. Nosotros, para agilizar la explicación, la asignaremos a mano.

```
$cadenaFecha = "2017-01-09";
```

Partiendo de esa cadena generaremos el objeto de la clase DateTime. Para esto tenemos dos opciones diferentes: utilizar procedimientos o bien mediante programación orientada a objetos.

a) Estilo por procedimientos

```
$fecha = date_create_from_format('Y-m-d',$cadenaFecha);
```

b) Estilo por objetos

```
$fecha = DateTime:: createFromFormat('Y-m-d',$cadenaFecha);
```

Utilizar una forma u otra es independiente. El resultado final será el mismo. Lo único que debemos tener claro es que a cada uno de estos métodos utilizado le hemos introducido dos parámetros:

1. El primero de ellos hace referencia al formato en el que estará la fecha original
2. La cadena o la variable que hace referencia a la fecha original

Para el primer parámetro se utiliza una notación que es conocida por la mayoría de los programadores y que consiste en el uso de una serie de letras que permiten indicar el formato en el que estará la fecha. Algunas de las opciones más importantes que podemos utilizar son:

- **d**: Día del mes representado con dos dígitos y con cero inicial. Del 01 al 31.
- **D**: Representación textual del día de la semana con tres letra.
- **j**: Día del mes sin cero inicial. Del 1 al 31.
- **l**: Representación textual completa del día de la semana.
- **w**: Representación numérica del día de la semana, del 0 al 6.
- **m**: Representación numérica de un mes, con cero inicial. Del 01 al 12.
- **M**: Representación textual corta de un mes, con tres letras.
- **n**: Representación numérica de un mes sin cero inicial. Del 1 al 12.
- **Y**: Representación numérica del año con cuatro dígitos.
- **y**: Representación numérica del año con dos dígitos.

Estas variables e pueden combinar para indicar diferentes tipos de formatos de fechas.

Convertir una fecha en cualquier formato de salida

index.php

```php
<?php
$cadenaFecha = "2017-01-09";

$fecha = DateTime:: createFromFormat('Y-m-d',$cadenaFecha);
echo $fecha->format("d/m/Y");

$fecha2 = date_create_from_format('Y-m-d',$cadenaFecha);
echo date_format($fecha2, "d-m-Y");
```

Una vez que tenemos ya nuestro objeto DateTime, podremos hacer numerosas cosas gracias a los distintos métodos que nos ofrece esta clase. En nuestro caso, lo que haremos será utilizar uno de esos métodos para transformar esa fecha en el formato que más nos interese.

Al igual que en el caso anterior, también lo podremos hacer de dos formas diferentes.

a) Por procedimientos
Para este caso lo que utilizaremos será la función "**date_format**", que recibirá dos parámetros. El primero de ellos será el objeto DateTime que hemos creado, mientras que el segundo corresponderá a la representación que queremos que tenga la fecha.

```
$fechaFormato = date_format($fecha, "d-m-Y");
```

Si imprimimos la variable por pantalla, obtendremos el resultado de **09-01-2017**.

b) Por objetos
Lo que utilizaremos será el método "**format**". En este caso, únicamente recibirá un parámetro que será la representación de la fecha de salida.

```
$fechaFormato = $fecha->format("d/m/Y");
```

En el ejemplo, si imprimimos la variable por pantalla el resultado sería **09/01/2017**.

Si nos fijamos en estas fechas hemos utilizado diferentes separadores. En uno ha sido la barra invertida mientras que otro ha sido el guión medio. Esto es independiente del método que utilicemos.

Como hemos visto a lo largo de este libro blanco, transformar el formato de una fecha en otra diferente resulta muy sencillo gracias al uso del objeto DateTime que nos ofrece PHP.