

# Bloque B

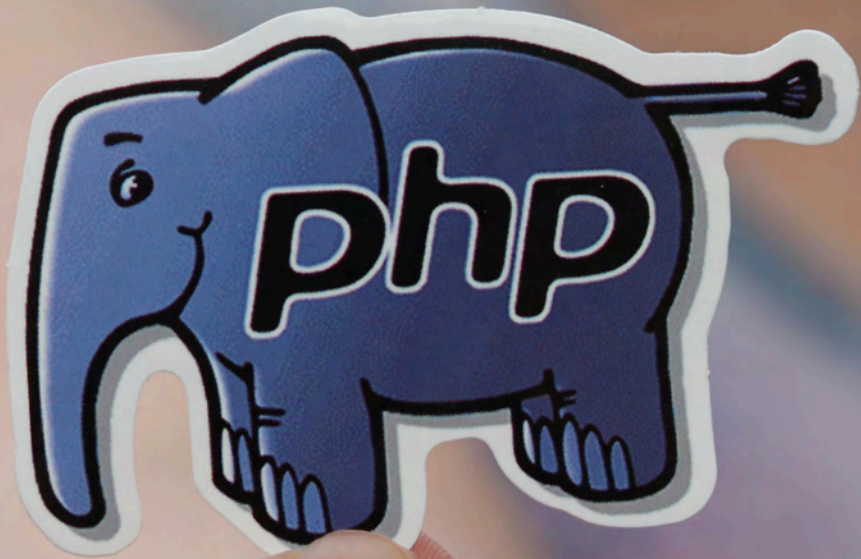
## Páginas Web Dinámicas

### Unidad 8. Fechas & Horas



# Contenidos

1. Introducción
2. Formatos de Fechas & Horas
3. Funciones integradas de fechas & horas
4. Objetos para representar fechas & horas
5. Uso de Intervalos
6. Generación de Eventos recurrentes
7. Gestión de diferentes Zonas horarias



# 1. Introducción

# Introducción

Las fechas y horas se pueden escribir de muchas maneras diferentes. PHP proporciona funciones y clases incorporadas que ayudan a procesar y mostrar fechas y horas en varios formatos.

# Introducción

En este capítulo, aprenderemos las diferentes formas en que el intérprete PHP puede aceptar fechas y horas como entrada, y cómo puede formatearlas como salida cuando son mostradas a los visitantes.

PHP puede trabajar con fechas y horas utilizando:

- Componentes como años, meses, días, horas, minutos y segundos
- Formatos como "1st June 2001", "1/6/2001" o "next Tuesday".
- Marcas de tiempo Unix (también conocido como *timestamp*) que cuentan el número de segundos desde el 1 de enero de 1970 (puede parecer una forma extraña de representar fechas/horas, pero muchos lenguajes de programación las utilizan)

# Introducción

Una vez que hayamos comprendido cómo procesa PHP los formatos de fecha y hora, conoceremos un conjunto de **funciones integradas que generan marcas de tiempo Unix (timestamps) y las convierten a un formato legible** por humanos.

Luego aprenderás como las fechas y horas pueden ser representadas utilizando objetos que son creados utilizando cuatro **clases incorporadas**:

- `DateTime` crea objetos que **representan una fecha y hora específicas**.
- `DateInterval` crea objetos que representan un **intervalo de tiempo** (por ejemplo, una hora o una semana).
- `DatePeriod` crea objetos que representan **eventos recurrentes** que ocurren a intervalos regulares (por ejemplo, cada día, mes o año)
- `DateTimeZone` crea objetos que representan una **zona horaria**





## 2. Formatos de Fechas & Horas

# Formatos de fecha

Las fechas pueden mostrarse de muchas maneras diferentes. PHP utiliza un conjunto de **caracteres de formato** (o **format characters**) para describir cómo se escribe una fecha.

Las fechas pueden constar de los siguientes componentes

- Día de la semana
- Día del mes
- Mes
- Año



# Formatos de fecha

PHP utiliza caracteres de formato para representar cada uno de estos componentes. Por ejemplo, `m-d-Y` representa el formato de fecha `04-06-2022`. Los caracteres de formato le indican al intérprete de PHP cómo serán las fechas:

- Procesadas cuando son recibidas
- Formateadas cuando se muestran

# Formatos de fecha

DAY OF WEEK		
CHARACTER	DESCRIPTION	EXAMPLE
<b>D</b>	First three letters	Sat
.....		
<b>l</b>	Full name	Saturday
DAY OF MONTH		
CHARACTER	DESCRIPTION	EXAMPLE
<b>d</b>	Digits with leading zero	09
.....		
<b>j</b>	Digits no leading zero	9
.....		
<b>S</b>	Suffix	th



# Formatos de fecha

## MONTH

CHARACTER	DESCRIPTION	EXAMPLE
<b>m</b>	Digits with leading zero	04
<b>n</b>	Digits no leading zero	4
<b>M</b>	First three letters	Apr
<b>F</b>	Full name	April

## YEAR

CHARACTER	DESCRIPTION	EXAMPLE
<b>Y</b>	Four digits	2022
<b>y</b>	Two digits	22

# Formatos de fecha

Se pueden agregar espacios, barras inclinadas, guiones y puntos entre los caracteres de formato para separar visualmente cada componente.

A continuación, puede ver cómo los caracteres de formato describen diferentes formas de escribir la misma fecha:

FORMAT CHARACTERS	DATE FORMAT
<b>l m j Y</b>	Saturday April 6 2022
<b>D jS F Y</b>	Sat 6th April 2022
<b>n/j/Y</b>	4/6/2022
<b>m/d/y</b>	04/06/22
<b>m-d-Y</b>	04-06-2022

# Formatos de tiempo

El tiempo puede constar de los siguientes componentes:

- Horas
- Minutos
- Segundos
- am/pm (si no se utiliza la hora de 24 horas)

# Formatos de tiempo

Cada uno de ellos puede representarse utilizando caracteres de formato. Por ejemplo, `g:i a` representa una hora en el formato `8:09 am`. Estos caracteres de formato se utilizan para indicar al intérprete PHP cómo serán las horas:

- Procesadas cuando se reciben
- Formateadas cuando se muestran



# Formatos de tiempo

## HOUR

CHARACTER	DESCRIPTION	EXAMPLE
<b>h</b>	12-hour with leading zero	08
<b>g</b>	12-hour no leading zero	8
<b>H</b>	24-hour with leading zero	08
<b>G</b>	24-hour no leading zero	8

## MINUTE

CHARACTER	DESCRIPTION	EXAMPLE
<b>i</b>	Digits with leading zero	09

# Formatos de tiempo

## SECOND

CHARACTER	DESCRIPTION	EXAMPLE
<b>s</b>	Digits with leading zero	04

## AM/PM

CHARACTER	DESCRIPTION	EXAMPLE
<b>a</b>	Lowercase	am
<b>A</b>	Uppercase	AM



# Formatos de tiempo

Se puede agregar espacios, dos puntos, puntos y paréntesis entre los caracteres de formato para separar visualmente cada componente.

A continuación, puedes ver cómo los caracteres de formato pueden describir diferentes formas de escribir la misma hora:

FORMAT CHARACTERS	DATE FORMAT
<b>g:i a</b>	8:09 am
<b>h:i(A)</b>	08:09(AM)
<b>6:i</b>	08:09

# Especificando fechas y horas mediante Strings

Algunas funciones y métodos permiten especificar una fecha y hora utilizando una cadena de caracteres (String). La cadena debe tener un formato aceptado, como se describe a continuación.

# Especificando fechas y horas mediante Strings

El intérprete PHP puede aceptar fechas utilizando los siguientes formatos de cadena. Si se utilizan barras inclinadas ( / ), el intérprete de PHP espera el mes antes del día del mes (formato estadounidense). Si se utilizan guiones ( – ) o puntos ( . ) como separadores, espera el día del mes antes del mes (formato europeo).

# Especificando fechas y horas mediante Strings

DATE FORMAT	EXAMPLE
<b>d F Y</b>	04 September 2022
<b>jS F Y</b>	4th September 2022
<b>F j Y</b>	September 4 2022
<b>M d Y</b>	Sep 04 2022
<b>m/d/Y</b>	09/04/2022
<b>Y/m/d</b>	2022/09/04
<b>d-m-Y</b>	04-09-2022
<b>n-j-Y</b>	9-4-2022
<b>d.m.y</b>	04.09.22



# Especificando fechas y horas mediante Strings

El intérprete PHP puede aceptar tiempos utilizando los siguientes formatos de cadena. Pero además, también puede:

- Utilizar mayúsculas o minúsculas para *am* y *pm*.
- Utilizar una letra `T` para separar una fecha de una hora. Ejemplo: `2023-10-15T14:30:00` se corresponde con el patrón `YYYY-MM-DDTHH:MM:SS`.
- Añadir una zona horaria a continuación.

# Especificando fechas y horas mediante Strings

12-HOUR TIME FORMAT	EXAMPLE
<b>ga</b>	4am
<b>g:i a</b>	4:08 am
<b>g:i:s a</b>	4:08:37 am
<b>g.i.s a</b>	4.08.37 am
24-HOUR TIME FORMAT	EXAMPLE
<b>H:i</b>	04:08
<b>H:i:s</b>	04:08:37
<b>His</b>	040837
<b>H.i.s</b>	04.08.37

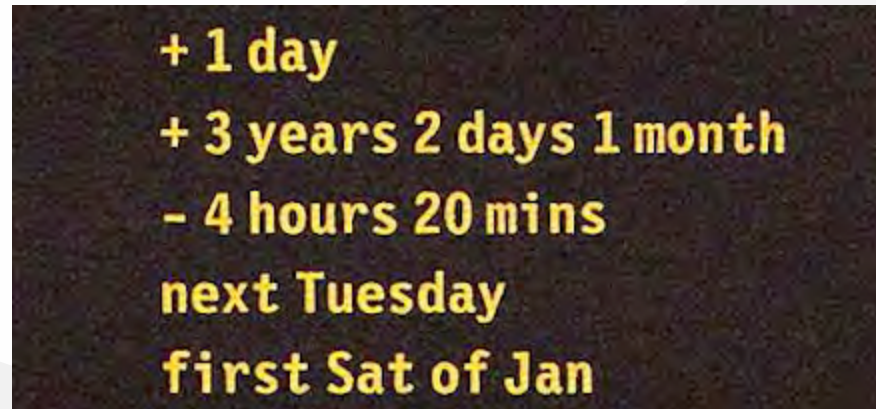


# Especificando fechas y horas mediante Strings

También se pueden usar los siguientes tiempos relativos:

TYPE	RELATIVE TIME
Add/subtract	<b>+</b> <b>-</b>
Quantity	<b>0 - 9</b>
Units of time (can be plural)	<b>day, fortnight, month, year, hour,</b> <b>min, minute, sec, second</b>
Day names	<b>Monday - Sunday</b> and <b>Mon - Sun</b>
Relative terms	<b>next, last, previous, this</b>
Ordinal terms	<b>first - twelfth</b>

# Especificando fechas y horas mediante Strings



+ 1 day  
+ 3 years 2 days 1 month  
- 4 hours 20 mins  
next Tuesday  
first Sat of Jan

`First` / `Last` sólo funciona para los días del mes.

Si no se especifica ninguna hora, se fijará a medianoche.

# Unix Timestamps

Las marcas de tiempo Unix (*timestamps*) representan **fechas y horas utilizando el número de segundos transcurridos desde la medianoche del 1 de enero de 1970.**

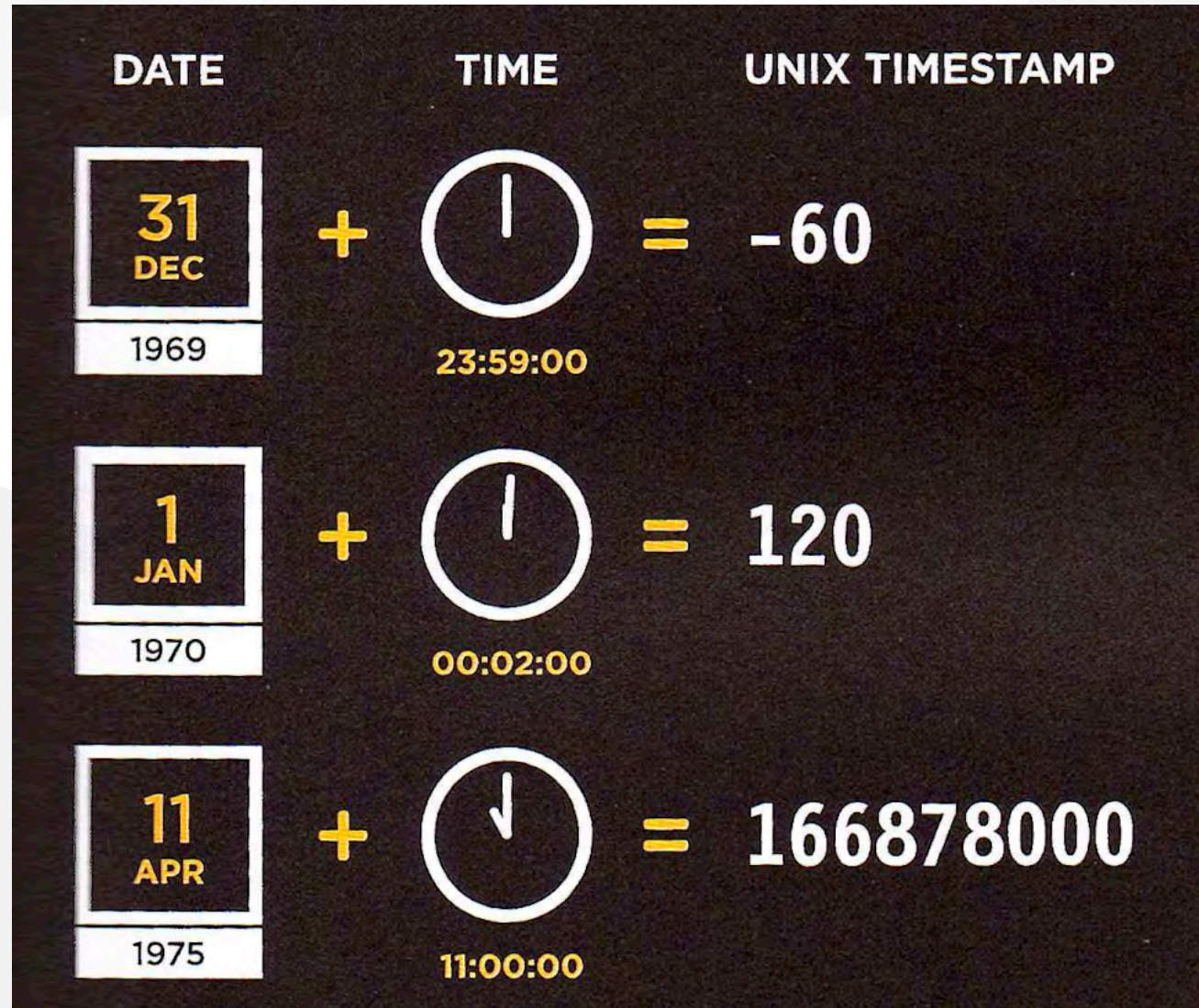
El intérprete PHP nos permite especificar y recuperar fechas y horas utilizando marcas de tiempo Unix.

# Unix Timestamps

En los siguientes ejemplos, se puede ver algunos ejemplos específicos de fechas y horas, seguidos de su correspondiente marca de tiempo Unix.

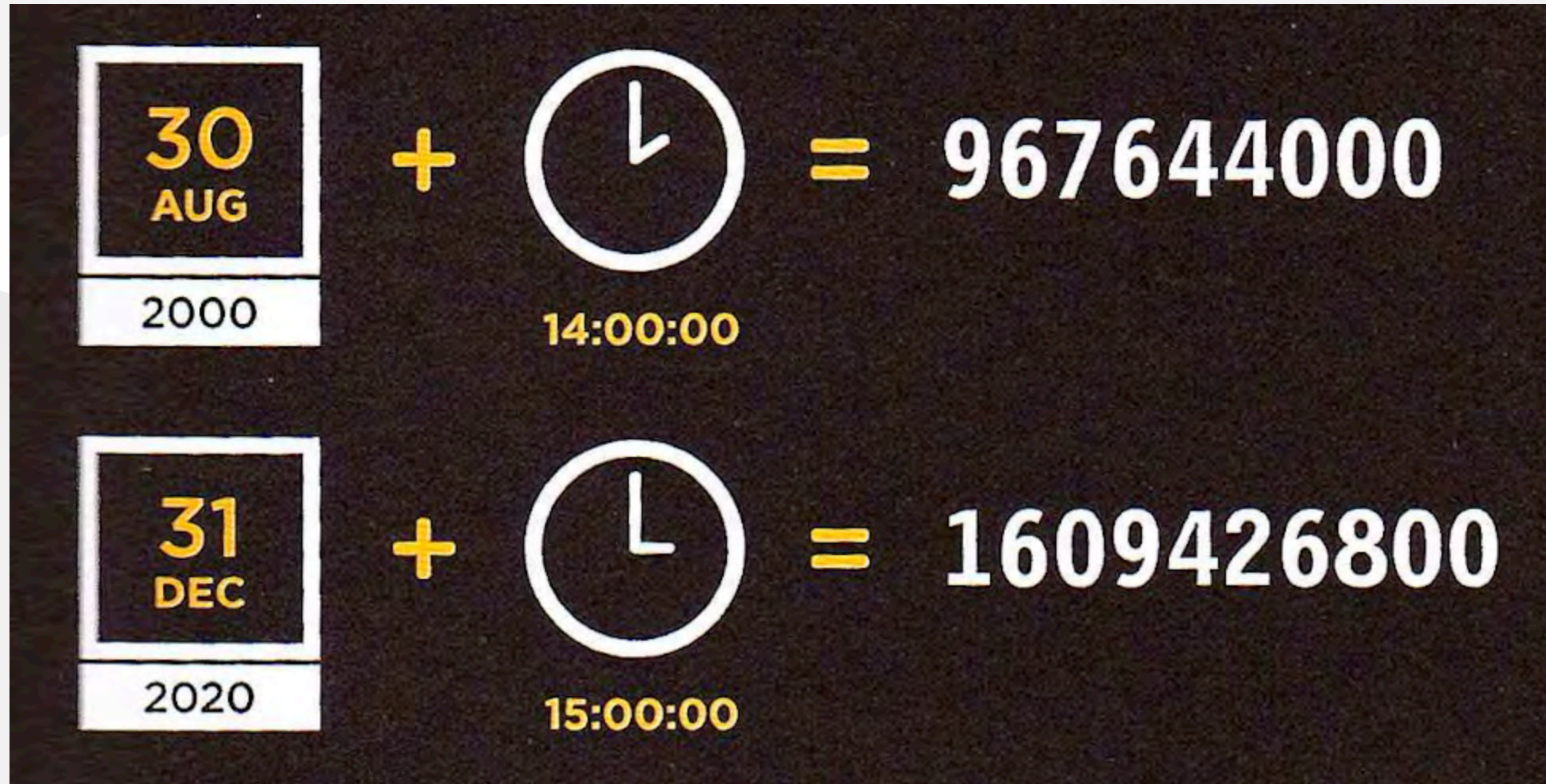


# Unix Timestamps





# Unix Timestamps



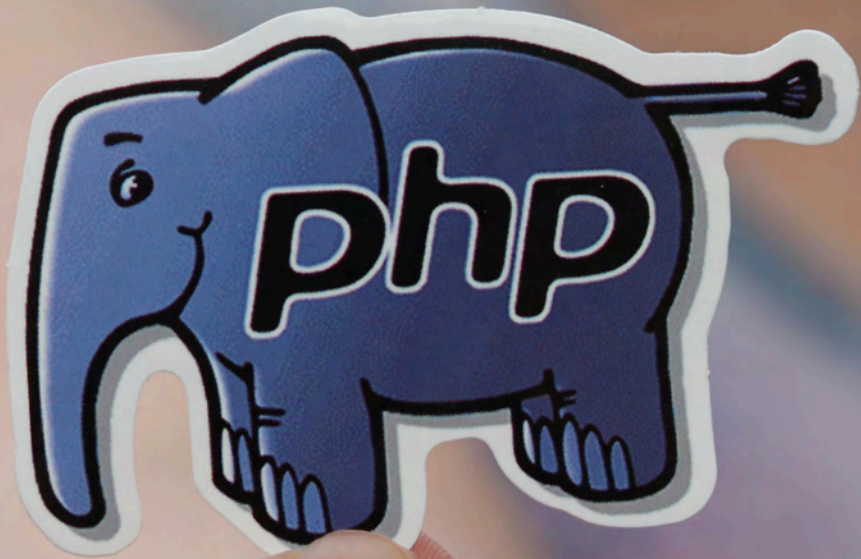
# Unix Timestamps

Las fechas **anteriores al 1 de enero de 1970** se escriben utilizando números **negativos**.

Como veremos a continuación, las funciones y clases incorporadas de PHP nos ayudan a trabajar con marcas de tiempo Unix. Utilizan los caracteres de formato que acabamos de conocer para describir como las funciones y clases deben **transformar una marca de tiempo Unix en algo que sea legible por humanos**.

La **fecha máxima** de una marca de tiempo Unix es el **19 de enero de 2038**.

Unix es un sistema operativo que se desarrolló en la década de 1970.



### 3. Funciones integradas de fechas & horas

# Funciones integradas de fecha y hora

PHP tiene funciones incorporadas que pueden **crear marcas de tiempo Unix para representar fechas y horas.**

También tiene funciones incorporadas para **convertir estas marcas de tiempo Unix en un formato que sea fácil de leer.**



# Funciones integradas de fecha y hora

Las tres funciones siguientes se utilizan para crear una marca de tiempo Unix.

Si no son capaces de crear una marca de tiempo, devuelven false.

Si no se especifica una hora para `strtotime()` o `mktime()`, la hora se establece a medianoche.



# Funciones integradas de fecha y hora

FUNCTION	DESCRIPTION
<code>time()</code>	Returns the current date and time as a Unix timestamp.
<code>strtotime(\$string)</code>	Converts a string to a Unix timestamp (accepts formats shown on p314).
	EXAMPLE
	<code>strtotime('December 1 2020');</code>
	<code>strtotime('1/12/2020');</code>
<code>mktime(H, i, s, n, j, Y)</code>	Converts date/time components (in arguments) into a Unix timestamp.
	EXAMPLE
	REPRESENTS
	<code>mktime(17, 01, 05, 2, 1, 2001);</code> February 1 2001 17:01:05
	<code>mktime(01, 30, 45, 4, 29, 2020);</code> April 29 2020 01:30:45

# Funciones integradas de fecha y hora

La función `date()` convierte las marcas de tiempo Unix a un formato legible por humanos.

El formato se especifica utilizando los **caracteres de formato** que hemos visto previamente en esta unidad.

Si no se indica ninguna marca de tiempo, se mostrará la **fecha y hora actuales**.

# Funciones integradas de fecha y hora

FUNCTION	DESCRIPTION
<code>date(\$format[, \$timestamp])</code>	Returns a Unix timestamp formatted in a human-readable way: The first parameter specifies how the date should be formatted. The second parameter is the Unix timestamp to format.
EXAMPLE	OUTPUT
<code>date('Y');</code>	Current year
<code>date('d-m-y h:i a', 1609459199);</code>	31-12-20 11:59 pm
<code>date('D j M Y H:i:a', 1609459199);</code>	Thu 31 Dec 2020 23:59:59

## Ejemplo: Funciones de fecha

A continuación vemos un ejemplo donde se calcula las fechas de inicio y fin de un periodo de ofertas (del 1 de enero al 1 de febrero de 2021), las formatea para mostrarlas con el día de la semana y el formato `d M Y`, e incluye una cabecera y pie de página desde archivos externos.



# Ejemplo: Funciones de fecha

1. La función `strtotime()` crea una marca de tiempo Unix para representar una fecha en el pasado. Se almacena en una variable llamada `$start`.
2. La función `mktime()` crea una marca de tiempo Unix para representar una fecha un mes más tarde. Se almacena en `$end`.
3. La función `date()` convierte estas marcas de tiempo Unix en un formato legible utilizando el:
  - Nombre del día
  - Día (con cero a la izquierda)
  - Mes (tres primeras letras)
  - Año (cuatro dígitos)

Estas se almacenan en las variables `$start_date` y `$end_date`.

## Ejemplo: Funciones de fecha

4. Se muestra la versión legible por humanos de cada fecha.
5. El archivo include para el pie de página añade un aviso de copyright. El año se escribe utilizando la función `date()`. Como no se indica ninguna marca de tiempo, utiliza la fecha actual.

# Ejemplo: Funciones de fecha

date-functions.php :

```
<?php
① $start      = strtotime('January 1 2021');
② $end        = mktime(0, 0, 0, 2, 1, 2021);
③ [ $start_date = date('l, d M Y', $start);
    $end_date   = date('l, d M Y', $end);
    ?>
    <?php include 'includes/header.php'; ?>

④ [ <p><b>Sale starts:</b> <?= $start_date ?></p>
    <p><b>Sale ends:</b> <?= $end_date ?></p>

    <?php include 'includes/footer.php'; ?>
```

includes/footer.php :

```
⑤ <footer>&copy; <?php echo date('Y')?></footer> ...
```

NOTA: Si la hora está desfasada unas horas, comprueba el ajuste de zona horaria por defecto en el archivo *php.ini*.

# Ejemplo: Funciones de fecha





# Actividad: Funciones de fecha

Vas a desarrollar un pequeño sistema para gestionar la **fecha de inicio y finalización de un evento** en un sitio web. El sistema permitirá al usuario ver las fechas actuales de los eventos, mostrar cuántos días faltan para que inicie o termine un evento, y ajustarse automáticamente según la fecha actual. También calcularás si el evento ya ha finalizado o está en curso.



# Actividad: Funciones de fecha

## Instrucciones:

1. **Fecha actual del sistema:** Utiliza `time()` para obtener la fecha y hora actual y formatearla usando `date()` para mostrar la fecha de hoy.

## 2. Fechas del evento:

- Define la **fecha de inicio** del evento con `strtotime()`. Usa una fecha que esté en el futuro.
- Define la **fecha de finalización** con `mktime()`, especificando manualmente el día, mes y año.



# Actividad: Funciones de fecha

## 3. Mostrar la cuenta regresiva:

- Calcula cuántos días faltan para que inicie el evento.
- Calcula cuántos días faltan para que termine el evento, desde la fecha actual.

## 4. Mensajes condicionales:

- Si la fecha actual es anterior al inicio del evento, muestra un mensaje que indique cuántos días faltan para que comience.
- Si la fecha actual está entre el inicio y el final del evento, muestra un mensaje indicando que el evento está en curso.
- Si el evento ya ha finalizado, muestra un mensaje indicando que ha concluido.





## 4. Objetos para representar fechas & horas



# Objetos para representar fechas y horas

La clase `DateTime` de PHP crea un **objeto que representa una fecha y hora**. Sus métodos pueden devolver la fecha y la hora que el objeto representa, ya sea en un formato legible por humanos o como una marca de tiempo Unix.

# Objetos para representar fechas y horas

Para crear un objeto `DateTime`, deberemos utilizar:

- Una variable que contenga el objeto
- El operador de asignación
- La palabra clave `new`
- El nombre de la clase `DateTime`
- Un par de paréntesis

# Objetos para representar fechas y horas

Entre paréntesis, añadiremos la **fecha/hora que debe representar el objeto**. Puedes utilizar cualquiera de los formatos de fecha y hora que se mostraron previamente en el apartado "*Especificando fechas y horas mediante Strings*" de esta unidad. El valor debe colocarse **entre comillas**.

# Objetos para representar fechas y horas

- Si no se especifica una fecha y hora, el objeto utilizará **la fecha y hora actuales**.
- Si se especifica una fecha pero no una hora, el objeto utilizará **la medianoche del día especificado**.

```
$date = new DateTime('2001-02-01 15:01:05');
```



The diagram illustrates the components of the code snippet above. Brackets are placed under each part of the code with labels below them: a bracket under '\$date' is labeled 'VARIABLE', a bracket under 'DateTime' is labeled 'CLASS NAME', and a bracket under the string '2001-02-01 15:01:05' is labeled 'DATE AND TIME'.



# Objetos para representar fechas y horas

También se puede utilizar la función `date_create_from_format()` para crear un objeto `DateTime`.

- El primer argumento es el formato en el que se suministrarán la fecha y la hora.
- El segundo argumento es la fecha y la hora en el formato especificado. Ambos argumentos van entre comillas.

```
$date = date_create_from_format('j-M-Y', '15-Jan-2020');
```

VARIABLE      FUNCTION      FORMAT      DATE/TIME

# Objetos para representar fechas y horas

Los siguientes métodos del objeto `DateTime` devuelven la fecha y la hora que representa el objeto.

- Para obtener la fecha/hora en un formato legible, utiliza el método `format()`.
- Para obtener la fecha/hora como una marca de tiempo Unix, utiliza el método `getTimestamp()`.

METHOD	DESCRIPTION
<code>format(\$format[, \$DateTimeZone])</code>	Gets the date and time in the specified format. The optional second parameter sets a time zone (see p326).
<code>getTimestamp()</code>	Returns the Unix timestamp for the date and time the object represents.

# Ejemplo: Objeto DateTime

1. Este ejemplo comienza creando un objeto utilizando la clase `DateTime` . Se almacena en una variable llamada `$start` .
2. Se crea un segundo objeto `DateTime` utilizando la función `date_create_from_format()` .
  - El primer parámetro especifica el formato en el que se proporcionará la fecha.
  - El segundo parámetro establece la fecha y la hora.
  - Este objeto se almacena en una variable llamada `$end` .

## Ejemplo: Objeto DateTime

3. La fecha y hora de inicio se escriben en la página utilizando el método `format()` del objeto `DateTime`. El argumento especifica el formato en el que deben escribirse la fecha y la hora.
4. La fecha (no la hora) final se escribe en la página utilizando el método `format()` del objeto `^`. Su parámetro especifica el formato en el que debe escribirse la fecha.
5. La hora final se escribe por separado y también utiliza el método `format()`. Esto muestra cómo se puede escribir simplemente la fecha o la hora que contiene el objeto.



# Ejemplo: Objeto DateTime

```
<?php
① $start = new DateTime('2021-01-01 00:00');
② $end   = date_create_from_format('Y-m-d H:i',
    '2021-02-01 00:00');
?>
<?php include 'includes/header.php'; ?>

<p><b>Sale starts:</b>
③ <?= $start->format('l, jS M Y H:i') ?></p>
<p><b>Sale ends:</b>
④ <?= $end->format('l, jS M Y') ?> <b>at</b>
⑤ <?= $end->format('H:i') ?></p>

<?php include 'includes/footer.php'; ?>
```

# Ejemplo: Objeto DateTime





# Actividad: Objeto DateTime

Eres un desarrollador web y estás creando un sistema para gestionar eventos en una página web. En este sistema, los usuarios pueden crear eventos futuros, y el sistema debe procesar las fechas de los eventos, almacenarlas y mostrarlas en diferentes formatos dependiendo de las necesidades del cliente.



# Actividad: Objeto DateTime

## Requisitos:

1. Los usuarios ingresan las fechas en un formato específico: `d/m/Y H:i:s` (por ejemplo, `16/10/2024 15:30:00` ).
2. El sistema debe convertir esta fecha en un objeto `DateTime` utilizando la función `date_create_from_format()` .
3. Una vez que se ha creado el objeto `DateTime` , el sistema debe:
  - Mostrar la fecha en formato: `Y-m-d H:i:s` (por ejemplo, `2024-10-16 15:30:00` ).
  - Obtener el *timestamp* UNIX correspondiente a esa fecha usando el método `getTimestamp()` .
  - Mostrar la fecha en formato legible como "16 de octubre de 2024, 15:30".