- Las expresiones regulares en PHP son una potente herramienta de programación que muchas veces se deja de lado ya que se cree que es muy compleja (aunque sí lo puede ser).
- Una expresión regular es una cadena especial que especifica un patrón para hacer búsquedas en otras cadenas. Si nuestra expresión regular encaja (hace "match") en la cadena la operación es exitosa
- El procesamiento de expresiones regulares es más costoso que usar funciones de búsqueda en cadenas (por ejemplo strstr), por lo que se debe considerar bien cuando utilizarlas y no usarlas en exceso.
- Para formar la cadena-patrón hay caracteres que tienen un significado especial en una expresión regular:

Metacaracteres	**.**	Match con cualquier carácter
	^	Match al principio del string
	$	Match al final del string
	\s	Match con cualquier espacio en blanco
	\d	Match con cualquier dígito
	\D	Match con cualquier carácter que no sea un dígito
	\w	Match con cualq. carácter que pueda ser parte de una palabra (letra, número, guión bajo)
	\W	Match con cualq. carácter que NO pueda ser parte de una palabra (letra, número, guión bajo)
	\A	Inicio de un string.
	\z	Final de un string.

Cuantificadores	*****	el carácter puede aparecer cero o mas veces.
	+	el carácter puede aparecer una o mas veces.
	?	el carácter puede aparecer cero o una vez.
	{n}	el carácter aparece exactamente n veces.
	{n,}	el carácter aparece n o más veces.
	{n,m}	el carácter puede aparecer entre n y m veces.

Agrupadores	**[ ]**	Permiten agrupar creando rangos. P.e. /ab[0-5]+c/ hará match con cualquier cadena que contenga ab, una o más veces un número entre 0 y 5, y finalmente una c. Por ejemplo: ab12c.
	( )	Permiten crear sub-expresiones, expresiones regulares contenidas dentro de otras: P.e. **/a(bc.)+e/**. Tiene un uso especial en formas como (…), que permite capturar todo lo que encierren los paréntesis, y (a|b) que hace match con a o b

- Las expresiones regulares se delimitan con el carácter **/** o con **#.**

  Por ejemplo, la expresion /ab?c/ hace match con **ac** y **abc**.

  la expresión /ab{1,3}c/ hace match con **abc**, **abbc**y **abbbc**.

- **Modificadores** Son caracteres que aparecen en la expresión regular después del delimitador de cierre y que permiten cambiar el modo en que se ejecuta la expresión regular.

  i Coincidir indistintamente entre mayúsculas y minúsculas.
  u Hacer los matches en modo UTF8
  x Ignorar espacios.

En PHP se dispone de la función **preg_match** para evaluar sí una cadena hace match con una expresión regular

preg_match(expr_regular , cadena)

Ej1. Validar formato de fecha (DD/MM/YYYY)

```
$patron = "/^\d{1,2}\/\d{1,2}\/\d{4}$/" ;

if ( preg_match( $patron , '5/10/2014') ) {
      echo " formato Ok ";
 }
 .....
```

Ej2. Validar direcciones de e_mail

```
function test_email($email) {
   $patron = "/^[a-zA-Z0-9]+([.][a-zA-Z0-9_]+)*@[a-zA-Z0-9_]+([.][a-zA-Z0-9_]+)*[.][a-zA-Z] {2,3}$/" ;

   if ( preg_match( $patron , $email) ) {
        return true;
    }
   return false;
 }
//----------------   Se examina la validez de algunos correos
//
$correos = array( 'luis@ab.com' , 'luis.garcia@ab.jz.es' , 'luis.@ab.com' , 'x@ab.jz..es');

for ($i=0;$i<4;$i++) {
       if ( test_email($correos[$i]) )  echo " el email $correos[$i] es correcto <br>";
       else  echo " el email $correos[$i] es erroneo <br>";
}
```

Ej3. Verificar el fomato de una IPv4

```
$uno   =  "([1-9]|[1-9]\d|1\d\d|2[0-4]\d|25[0-5])";
$resto =  "([.](\d|[1-9]\d|1\d\d|2[0-4]\d|25[0-5]){3}";

$patron=  "/^" . $uno  . $resto . "$/" ;

 if ( preg_match( $patron , '10.1.160.200') ) {
       echo " IP Ok ";
   }
   .....
```