Desarrollo de Aplicaciones en Internet

Universidad Autónoma de Zacatecas

PHP

Instructor: Dr. Roberto Solís Robles

PHP BÁSICO

Breve Historia de PHP

PHP (PHP: Hypertext Preprocessor) fue creado por Rasmus Lerdorf en 1994. Fue inicialmente desarrollado para uso en generación de bitácoras de uso de HTTP y generación de formas del lado del servidor en Unix.

PHP 2 (1995) transformó el lenguaje en un lenguaje de scripts incrustados del lado del servidor. Se agregó soporte para bases de datos, cargas de archivos, variables, arreglos, funciones recursivas, condicionales, ciclos, expresiones regulares, etc.

PHP 3 (1998) agregó soporte para fuentes de datos ODBC, soporte multiplataforma, protocolos de email (SNMP,IMAP), y un nuevo analizador sintáctico escrito por Zeev Suraski y Andi Gutmans.

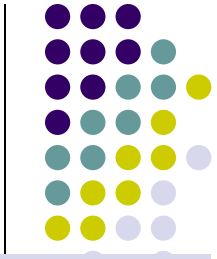
PHP 4 (2000) se convirtión en un componente independiente del servidor web para una mejor eficiencia. El analizador fue renombrado como Zend Engine. Se agregaron muchas características de seguridad.

PHP 5 (2004) agrega el Zend Engine II con POO, soporte de XML usando la librería libxml2, extensiones SOAP interoperabilidad con servicios Web.

¿Qué se requiere para trabajar con PHP?

- Si su servidor web soporta PHP
  - No necesita nada
  - Simplemente cree algunos archivos.php en su directorio web
- Si su servidor web no soporta PHP, debe instalar PHP.
  - Opción 1
    - Baje PHP
    - Baja un manejador de base de datos(MySQL)
    - Baje un servidor web (Apache)
  - Opción 2:
    - Baje un paquete con todos los elementos anteriores (XAMPP, LAMP, WAMP, MAMP, etc.)

¿Por qué se usa PHP? (1)

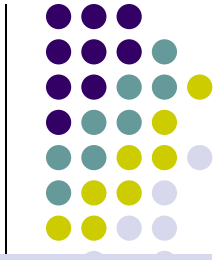


1. Fácil de usar
   El código es incrustado en HTML. El código PHP se encierra en etiquetas especiales de inicio y terminación que le permiten entrar y salir de "modo PHP".

…

```
<html>
    <head>
        <title>Ejemplo</title>
    </head>
    <body>

        <?php
          echo "Hola, soy un texto escrito con PHP!";
          ?>

    </body>
</html>
```

¿Por qué se usa PHP? (2)



2. Es multiplataforma
   Se ejecuta en la casi cualquier servidor web en varios sistemas operativos.
   Una de las mas grandes características es el amplio rango de bases de datos soportadas

- Servidores Web : Apache, Microsoft IIS, Caudium, Netscape Enterprise Server
- Sistemas Operativos: UNIX (HP-UX,OpenBSD,Solaris,Linux), Mac OSX, Windows NT/98/2000/XP/2003/Vista/7
- Bases de datos soportadas: Adabas D, dBase,Empress, FilePro (solo lectura), Hyperwave,IBM DB2, Informix, Ingres, InterBase, FrontBase, mSQL, Direct MS-SQL, MySQL, ODBC, Oracle (OCI7 y OCI8), Ovrimos, PostgreSQL, SQLite, Solid, Sybase, Velocis, Unix dbm

¿Por qué se usa PHP? (3)

3. Costo Beneficio

PHP es gratuito. Código fuente abierto significa que la comunidad completa de PHP contribuirá a arreglar defectos. Existen varias tecnologías add-on (librerías) para PHP que también son gratuitas.

	PHP
Software	Gratuito
Plataforma	Gratuita (Linux)
Herramientas de Desarrollo	Gratuitos PHP Coder, jEdit

Comenzando con PHP (1)

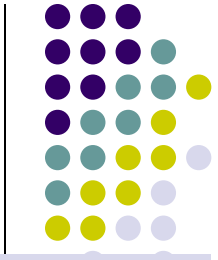
1. **Entrando y saliendo de modo PHP**

- **PHP analiza un archivo buscando etiquetas que le dicen que debe empezar a interpretar el texto como código PHP. El analizador ejecuta entonces todo el código que encuentra hasta que se topa con una etiqueta de cierre PHP.**

<?php echo “Hola Mundo”; ?>

Etiqueta de Inicio	Etiqueta de Cierre	Notas
<?php	?>	Método preferido ya que permite el uso PHP con XHTML
<?	?>	No recomendado. Más fácil de escribir, pero tiene que ser habilitado y puede entrar en conflicto con XML
<script language="php">	</script>	Siempre disponible, mejor opción si es usado con FrontPage como editor HTML
<%	%>	No recomendado. Soporte para etiquetas ASP fue agregado en versión 3.0.4

Comenzando con PHP (2)



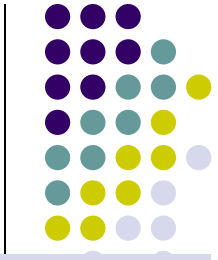
2. **Página HTML Simple con PHP**

- **El siguiente es un ejemplo básico para generar texto de salida usando PHP.**

```
<html><head>
<title>Mi Primer Pagina PHP </title>
</head>
<body>
<?php
   echo "Hola Mundo!";
?>
</body></html>
```

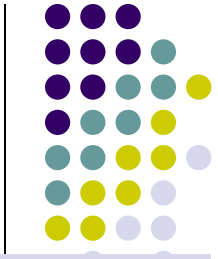
Note que el punto y coma es usado al final de cada línea de código PHP para indicar un final de línea. Como en HTML, PHP ignora los espacios en blanco entre líneas de código.

Fundamentos de la Sintáxis de PHP



- El bloque de script comienza con **<?php** y termina con **?>**
- Cada línea de código en PHP debe terminar con un **;**
- Comentarios
  - *// ,# comentario*
  - */* comentario **/*
- Escritura de texto plano
  - **echo** "*texto*"
  - **print** "*texto*"

Variables



- Cada variable inicia con el símbolo **$**
- El nombre de una variable puede contener solo a-Z,0-9,_
- No necesita estar declarada antes de darle un valor.

```
<?php
        $txt = "Hola Mundo!";
        $numero = 16;
 ?>
```

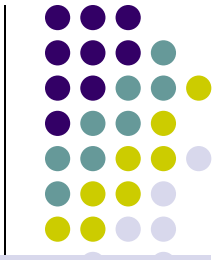
Constantes

- Son valores que nunca cambian
- Se definen en PHP usando la función define.

```
define("UAZ", "Universidad Autónoma de Zacatecas")
```

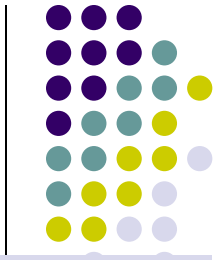
- La función defined(constante) indica si la constante existe o no.

Tipos de Datos



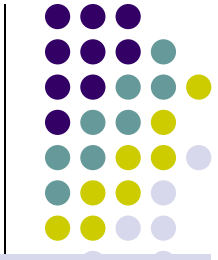
- Numéricos
  - Integer – positivos y negativos, incluyendo 0
  - Float – números reales con 14 dígitos de exactitud
- Lógicos
  - Boolean - True y False, sin diferencia entre mayus/minus
- Alfabéticos
  - String – conjunto de caracteres
- Otros Tipos
  - Array
  - Object
- PHP es un lenguaje sin tipos (una variable puede cambiar de tipo de dato dinámicamente)

Trabajando con Variables



- **settype($var, "*integer*")**
  - Le permite establecer una variable al tipo deseado
- **gettype($var)**
  - devuelve el tipo de la variable
- **.**
  - Conecta 2 variables de tipo string
- **strlen($var)**
  - devuelve la longitud de un string
- **isset ($var), unset($var), is_int ($var), etc.**

Operadores en PHP



- Operadores matemáticos **+,-,/,*,%**
  - **++** (**--**) para incremento (decremento) por 1
- Operadores de comparación **>,<, >=,==, !=**
- Operadores de asignación
  - **x+=y** equivalente a x=x+y
  - **x-=y** equivalente a x=x-y
  - **x*=y** equivalente a x=x*y
  - **x/=y** equivalente a x=x/y
  - **x%=y** equivalente a x=x%y
- Operadores lógicos: **&&=and, ||=or, !, xor**

Sentencias Condicionales

- **if/ elseif/ else**

```
if ($i == 0) {
    echo "i igual a 0";
} elseif ($i == 1) {
    echo "i igual a 1";
} elseif ($i == 2) {
    echo "i igual a 2";
} else {
    echo "i no es 0, 1 o  2";
}
```

- **switch**

```
switch ($i) {
case 0:
    echo "i igual a 0";
    break;
case 1:
    echo "i igual a 1";
    break;
case 2:
    echo "i igual a 2";
    break;
default:
    echo "i no es 0, 1 o  2";
}
```

Ciclos

- while

```
$i=1
while ($i <= 10) {
    echo $i;
    $i++;
}
```

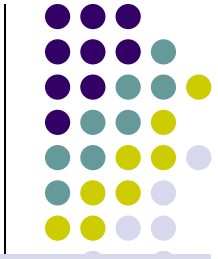
- do …while

```
$i=1
do {
    echo $i;
    $i++;
} while ($i < 10);
```

- for

```
for ($i=1; $i<=10; $i++) {
    echo $i;
}
```

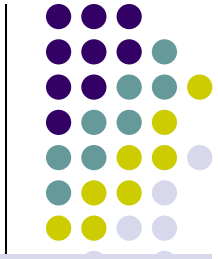
Arreglos



- Arreglos numéricos
  - Cada elemento del arreglo tiene su índice (iniciando en 0)
  - **$*nombres* = array(**"*Pedro*","*Jose*"**);**
  - **$*nombres*[0] = "*Pedro*";**

- Arreglos asociativos
  - Cada elemento se identifica con una llave
  - **$*edades* = array("*Pedro*"=>*20*, "**Jose**"=>*34*);**
  - **$*ages*['*Pedro*'] = *20*;**

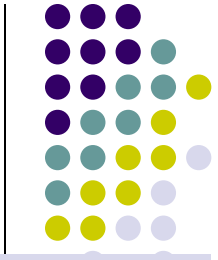
Arreglos Multidimensionales



- El elemento de un arreglo es a su vez un arreglo

```
$familias = array
    (
    "Solis"=>array
        ( "Roberto", "Alberto", "Sahara" ),
    "Rivera" =>array
        ("Pedro", "Paulina" )
    )
```

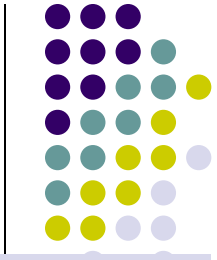
Ciclo foreach



- Se puede usar el ciclo foreach para iterar sobre los elementos de un arreglo:

```
$colores = array('rojo','azul', 'verde');

foreach ($colores as $color) {
    echo $color;
}
```

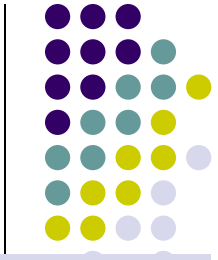
Funciones (1)



- Toda función comienza con **function($*parametros*)**
- Los requerimientos para nombrar funciones son los mismos que para nombrar variables
- El símbolo **{** abre el código de la función, y el símbolo **}** lo cierra
- Los parametros son opcionales, si son varios, se separan por comas

- Mas de 700 funciones internas disponibles

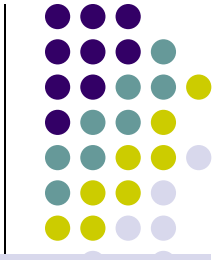
Funciones (2)



- Se pueden asignar funciones a las variables
  - Por ejemplo
    - $funcvar = “mifuncion”;
    - Para invocar la funcion mifuncion () usando la variable:
      $funcvar( );
- Cuando un argumento se quiere pasar por referencia, se coloca un ampersand (&) antes del nombre del parámetro
  - Por ejemplo:

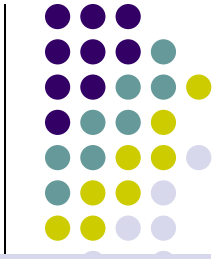
    mifuncion(&$mivarporreferencia);

Funciones Internas (1)



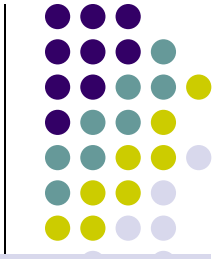
- Funciones de Manipulación de Arreglos
  - sort, merge, push, pop, slice, splice, keys, count
- Funciones COM : Interface a objetos COM de Windows
- Funciones de fecha y hora
  - getdate, mkdate, date, gettimeofday, localtime, strtotime, time
- Funciones de Directorio
  - Independiente de la plataforma

Funciones Internas (2)



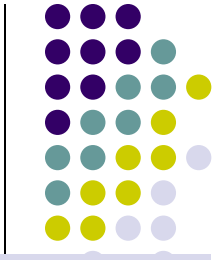
- Funciones de manejo de errores
  - Para recuperarse de advertencias y errores
- Funciones del Sistema de Archivos
  - Acceso a archivos regulares
  - Verificar información de status de directorios y archivos
  - Copiar, borrar y renombrar archivos
- Funciones de Correo Electrónico
  - mail($recipient, $subject, $message)

Funciones Internas (3)



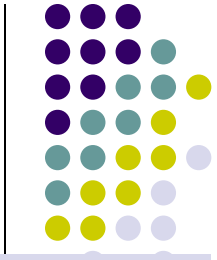
- Funciones de Bases de Datos
  - MySQL, Oracle, PostgreSQL, SQL Server, etc.
- PDF
  - Crear/manipular archivos PDF dinámicamente
- Funciones de Expresiones Regulares
- Funciones de Administración de Sesiones
- etc. etc.

Alcance (Scope)



- Las funciones tienen un alcance global, las funciones definidas dentro de funciones están disponibles en todas partes
- Las variables asignadas o modificadas dentro de una inside of a función tienen su valor sólo dentro de la función
- Sin embargo, se puede declara que una variable usada dentro de una función sea considerada como global, pero en general, es mejor dejar el alcance de las variables limitado, de tal forma que la función sea portable

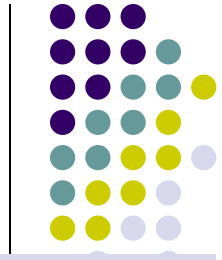
Las funciones tienen un alcance global



- A diferencia de las variables, las funciones son globales en su alcance
- Pero una función no se ejecuta sino hasta ser llamada
- Aquí, externa() es usada para crear interna()
- Una vez que externa() es llamada, interna() existe
- interna() esta disponible fuera de externa ()

```
function externa() {
   function interna() {
     echo "No existo hasta que externa () sea llamada.\n";
   }
}

/* No podemos llamar a interna() aun pues no existe. */

externa();

/* Ahora podemos llamar a interna (), la ejecucion de
externa() la ha hecho accesible. */
interna();

?>
```

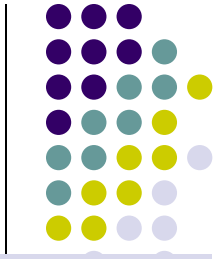
Funciones y valores por default



```
function hazHeading($elemento, $nivel="2")
   {
      echo "<h" . $nivel . ">$elemento</h" . $nivel . ">\n";
   }
function hazHeadingIncorrecto($nivel="2", $elemento)
   {
      echo "<h" . $nivel . ">$elemento</h" . $nivel . ">\n";
   }

hazHeading("Universidad Aut Zac");
hazHeading("UAZ", 1);
hazHeadingIncorrecto(2, "No funciona");
hazHeadingIncorrecto("A que tampoco este funciona");
```

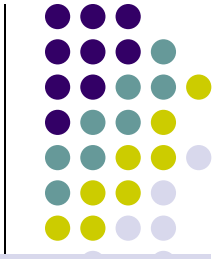
NOTA: Los valores por default tienen que estar en los parametros a la derecha (ultimos parametros)

Inclusión de Archivos



- Es posible dentro de un archivo .php incluir el contenido de otros archivos (que no necesariamente tienen que ser .php)
- Para ello, se usa la sentencia include, por ejemplo:

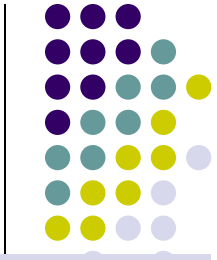
```
include "archivo.html"
```



PHP

MANEJO DE FORMAS Y SESIONES

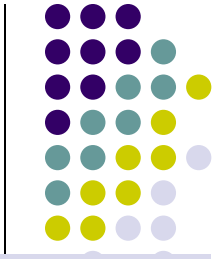
Formas y Entrada del Usuario



- Se puede acceder a la información que un usuario teclea en una forma XHTML

```
<html>
<body>
<form action="bienvenido.php" method="post">
   Nombre: <input type="text" name="nombre" />
   Edad: <input type="text" name="edad" />
   <input type="submit" />
</form>
</body>
</html>
```

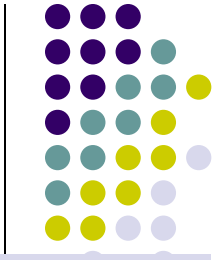
Arreglo $_GET



- Usado para recolectar los valores de una forma
- Los nombres de variables y valores correspondientes están en el URL
  - http://localhost/bienvenido.php?nombre=jose&edad=39
- Se puede enviar una cantidad limitada de información(máximo 100 caracteres)

```
<html>
<body>
   Bienvenido <?php echo $_GET["nombre"]; ?> <br />
   Tu edad es <?php echo $_GET["edad"]; ?>.
</body>
</html>
```

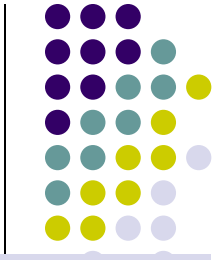
Arreglo $_POST



- Usado para recolectar los valores de una forma
- La información de una forma no es visible en el URL
  - http://localhost/bienvenido.php
- No hay limite en cuanto a la cantidad de información que puede ser enviada

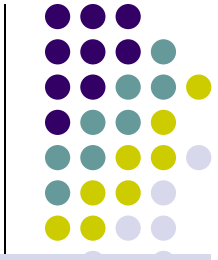
```
<html>
<body>
  Bienvenido <?php echo $_POST["nombre"]; ?><br />
  Tu edad es <?php echo $_POST["edad"]; ?>.
</body>
</html>
```

Sesiones (1)



- Un aspecto muy útil de PHP es la capacidad de mantener datos de la sesión
- Para hacer esto, se crea una sesión y se carga con datos
- Los datos de la sesión se almacenan del lado del servidor, se le asigna una clave(pero se encripta), y permanece para el URL y direccion IP remota

Sesiones (2)



- Las sesiones necesitan ser creadas antes de que los encabezados HTML sean enviados, por tanto esto tiene que hacerse al principio del código PHP
- Use $_SESSION[nombrevar] para crear espacios para una variable de nombre nombrevar
- Después de que las variables hayan sido llenadas con datos, se puede accesar a ellas usando el arreglo $_SESSION
- En el siguiente ejemplo definiremos una clave (generada aleatoriamente) y un contador

Ejemplo de Manejo de Sesión (1) -- manejosesion.php

```
<?php
// Iniciamos una sesion, y establecemos una clave y un contador
// Una sesion es un metodo de rastrear el estado de una sesión
// dada del navegador, y permite almacenar datos en el servidor
        session_start(); // creamos la sesion
        // usaremos una variable de sesion para una clave para
        // rastrear la sesion y asi poder detectar recargas
        // y un contador de sesion usado para determinar en que
        // paso vamos en la sesion
```

Ejemplo de Manejo de Sesión (2)

Ahora podemos checar si la clave esta establecida, y si no lo está, es la primer carga de la página por lo cual generamos una clave y establecemos el contador a 0….

```
// si la clave de sesion no esta establecida, generamos un
// numero aleatorio como clave y establecemos el contador a 0
// (esta situacion indica que la forma esta siendo cargada por
// primera vez)
if (!$_SESSION['clave_sesion']) {
   $_SESSION['clave_sesion'] = rand();
   $_SESSION['contador_sesion'] = 0;
}
```

Ejemplo de Manejo de Sesión (2)

y podemos entonces construir la forma (note el uso de PHP_SELF y el input oculto) …

```
// Si no tenemos clave, entonces el usuario no ha llenado la forma
if (!$_GET["numero_clave"]){
   echo "<form action=\"" . $_SERVER["PHP_SELF"] .
      "\" method=\"GET\" name=\"opciones\">\n";
   echo "<p>Pelicula favorita?</p><p><textarea rows=\"1\"
cols=\"50\" wrap=\"virtual\" name=\"respuesta\"></textarea></p>\n";
   echo '<input type="hidden" value="' .
         $_SESSION['clave_sesion'] . '" name="numero_clave" />';
   echo '<p><input type="submit" value="Enviar" /></p>';
   echo '</form>';
   $_SESSION['contador_sesion'] = 1;
}
```

Ejemplo de Manejo de Sesión (3)

Posteriormente, tenemos un elseif que determina si tenemos los datos que necesitamos, que la llave exista y que el contador este en 1 Si es asi, imprimos el valor de las variables recibidas …..

```
} elseif ($_SESSION['clave_sesion'] == $_GET["numero_clave"] &&
$_SESSION['contador_sesion'] == 1 && $_GET["respuesta"]) {
   echo "Este es el arreglo GET :";
   echo "<pre>";
   print_r($_GET);
   echo "</pre>";
```

Ejemplo de Manejo de Sesión (4)

… generamos una forma de confirmación y establecemos el contador a 2

Nuevamente hacemos que la forma se envíe a este mismo script

```
  echo "<form action=\"" . $_SERVER["PHP_SELF"] . "\"
                method=\"GET\" name=\"opciones\">\n";
  echo '<input type="hidden" value="' . $_SESSION['clave_sesion'] .
                '" name="numero_clave" />';
  echo '<p>Los aceptas?</p>';
  echo '<input type="radio" value="si" name="si_no" />SI' .
         ' <input type="radio" value="no" name="si_no" />NO';
  echo '<p><input type="submit" value="Enviar" /></p>';
  echo '</form>';
  $_SESSION['contador_sesion'] = 2;
}
```

Ejemplo de Manejo de Sesión (5)

Continuamos ahora verificando la confirmación enviada y terminamos la sesión, haciendo lo que corresponda por haber terminado exitosamente

```
//Verifique nuevamente que  clave_sesion concuerde con
// numero_clave, y que estamos en el paso 2, y se haya dado  "si"
elseif ($_SESSION['clave_sesion'] == $_GET["numero_clave"] &&
$_SESSION['contador_sesion'] == 2 && $_GET["si_no"] == "si") {
   echo '<p>Excelente, todo va bien.</p>';
   echo '<h1>Hemos terminado sin error</h1>';
   // Destruye la sesion
   session_unset();
   session_destroy();
}
```

Ejemplo de Manejo de Sesión (6)

Checamos si no se confirmo la información mostrada

```
// Verifique nuevamente que  clave_sesion concuerde con
numero_clave, y que estamos en el paso 2, y se haya dado "no"
elseif ($_SESSION['clave_sesion'] == $_GET["numero_clave"] &&
$_SESSION['contador_sesion'] == 2 && $_GET["si_no"] == "no") {
   echo '<p>Sorry, algo estuvo mal, <a href="manejosesion.php">' .
        'intentas de nuevo?</a></p>';
   echo '<h1>Hubo un errorsote</h1>';
   // Destruye la sesion
   session_unset();
   session_destroy();
}
```

Ejemplo de Manejo de Sesión (7)

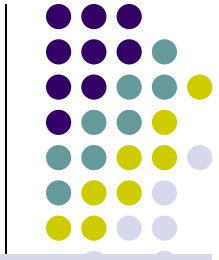
Y finalmente, indicamos que hacer si alguna de las condiciones no se cumplio. Debemos hacer notar que esta verificación esta imcompleta pues no estamos determinando la causa específica del error

```
// Si no se cumple alguna condicion, asumimos que algo estuvo mal y
// reiniciamos
else {
   echo '<p>Lo siento, encontre un problema, probablemente no ' .
        'llenaste algun campo. <a href="manejosesion.php"> ' .
        'Continuamos desde el principio?</a></p>';
   // Destruye la sesion
   session_unset();
   session_destroy();
}
?>
```

PHP

EXPRESIONES REGULARES

Sintaxis de ER

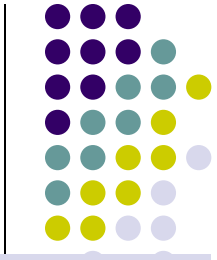


- Dos principales dialectos, POSIX y Perl (nos enfocaremos en el primero)
- El uso básico de una ER es “concardar patron en texto”, donde el patron es lo que estamos buscando y texto es donde pudiera estar lo que buscamos
- Hay muchas variantes para hacers búsquedas y reemplazos, pero la función básica es ereg()

Ejemplos de ereg()

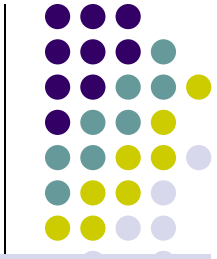
- ereg("feo", $texto) buscamos el string "feo"
- ereg([0-9], $texto) buscamos cualquier instancia de un digito de 0 a 9
- ereg([cq], $texto) buscamos una **c** o una **q** (similar a buscar "c|q"
- ereg(([0-3])|(feo), $texto) buscamos un texto compuesto por un digito entre 0 y 3 o que sea igual a "feo" (el | es un or)
- ereg((^Esto), $texto) busca el string “Esto” al principio de $texto

Mas ejemplos de ereg()



- ereg((Esto$), $texto) busca el string “Esto” al final de $texto
- ereg([7-9].ing, $texto) busca un texto que comience con 7, 8 o 9, seguido de cualquier caracter y el string "ing" (el punto representa cualquier caracter)
- ereg(rx{4,7}, $texto) busca un string que comience por r y sea seguido por 4 a 7 x's (las llaves indican un rango aplicado al caracter anterior)
- ereg([0-9]{5}s, $texto) busca 5 digitos de 0 a 9, seguidos por una s

Quitando el significado a los caracteres especiales en ER

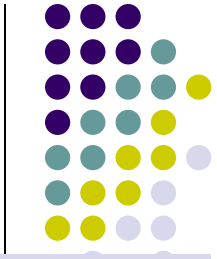


- ^.[$()|*+?{\ son caraceteres especiales en las ER, y se les quita su significado anteponiendoles una diagonal invertida
- ereg(\[@@@\], $texto) busca el string '[@@@]'
- ereg('',\.\[@@, $texto) busca el string '',.[@@'

PHP

MYSQL

Acceso a Bases de Datos MySQL



- PHP proporciona una gran funcionalidad para acceso a bases de datos de diverso tipo
- Nos enfocaremos en el acceso a bases de datos MySQL
- Usaremos dos scripts PHP para ejemplificar el uso de bases de datos MySQL:
  - accesomysql.php
  - muestratabla.php

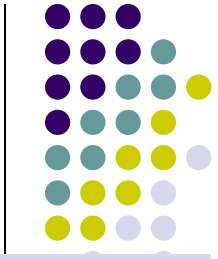
accesomysql.php

- Este script realizará los siguientes pasos:
  - Conectarse a la base de datos encontrada en la computadora indicada por la variable $hostbd, y con el usuario y clave indicados por las variables $usuariodb y $clavedb
  - Seleccionar la base de datos indicada por la variable $nombredb y obtener una lista de las bases de datos que contiene
  - Construir una forma donde se mostraran las tablas encontradas en un drop down list, de donde el usuario podra seleccionar una tabla
  - Al dar click en Enviar en esta forma, se enviara el nombre de la tabla seleccionada al script muestratabla.php

muestratabla.php

- Este script realizará las siguientes tareas:
  - Recibir como parámetro el nombre de la tabla seleccionada en el script accesomysql.php
  - Conectarse a la base de datos encontrada en la computadora indicada por la variable $hostbd, y con el usuario y clave indicados por las variables $usuariodb y $clavedb
  - Seleccionar la base de datos indicada por la variable $basedatos y seleccionar todos los registros de la tabla recibida como parámetro
  - Generar una tabla que contenga todos los registros y columnas obtenidos de la tabla seleccionada

Funciones Cubiertas



- mysql_connect()
- mysql_error()
- mysql_select_db()
- mysql_query()
- mysql_num_rows()
- mysql_fetch_array()
- mysql_fetch_row()
- mysql_num_fields()
- mysql_fetch_field()
- mysql_free_result()
- mysql_close()